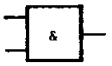
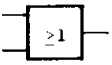
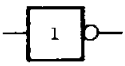
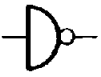
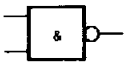
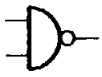
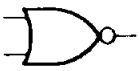
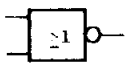
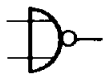
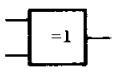
Lisbeth Dragsted
Niels Jørgen Vestergaard
Niels Henrik Würtz

Computerens Fysik

eksperimentel digitalteknik

F&K Forlaget

Oversigt over gates

Navn og IC	US	IEC	DIN	Funktions-tabel	Logisk funktion
AND 7408				A B | A ∧ B L L | L L H | L H L | L H H | H	$A \wedge B$
OR 7432				A B | A ∨ B L L | L L H | H H L | H H H | H	$A \vee B$
NOT 7404				A | $\overline{A}$ L | H H | L	$\overline{A}$
NAND 7400				A B | $\overline{A \wedge B}$ L L | H L H | H H L | H H H | L	$\overline{A \wedge B}$
NOR 7402				A B | $\overline{A \vee B}$ L L | H L H | L H L | L H H | L	$\overline{A \vee B}$
EX-OR 7486				A B | EXOR L L | L L H | H H L | H H H | L	$\overline{A \Leftrightarrow B}$

Lisbeth Dragsted
Niels Jørgen Vestergaard
Niels Henrik Würtz

Computerens Fysik

eksperimentel digitalteknik

F&K Forlaget

COMPUTERENS FYSIK
af Lisbeth Dragsted, Niels Jørgen Vestergaard
og Niels Henrik Würtz

© F&K Forlaget
Slotsgade 2[3]
2200 København N

Tilrettelæggelse: Steen Hoffmann
Omslag: Jacques Chevallier
Tegninger: Tove Asmussen
Fotograf: Ivar Mjell

Trykt hos Budolfi Tryk, Aalborg
Mekanisk, fotografisk eller anden gengivelse
af denne bog eller dens enkelte dele er
ikke tilladt uden lovlig hjemmel.

ISBN 87-87229-01-3

1. udgave 1985.

Forsiden viser hukommelsesenhederne i EPROM'en 27C64 af typen CMOS. Forstørrelsen, der er på ca. 2000 gange, er lavet med et scanningelektronmikroskop på Fysisk Institut, Århus Universitet, af Jacques Chevallier.
En EPROM benyttes i forbindelse med en computer, hvor den indeholder opstarts- og standardprocedurer til processoren. 27C64 indeholder 8K × 8 eller 65536 enheder (nuller eller éttaller). Enhederne er inden i IC'en placeret på overfladen af en siliciumkrystal, der måler 5 × 5 mm^2, og med forstørrelse på forsiden svarer det til areal på 100 m^2.

Indholdsfortegnelse

Forord

Arbejdet med denne bog tog sit udgangspunkt i en workshop på Det Fysiske Institut ved Århus universitet. Vi ønskede selv at vide noget om den digitalteknik, der trænger sig frem overalt.
Med bogen forsøger vi at vise, at digitalteknikken bygger på simple fysiske love og har derfor gjort meget ud af at vise, hvorledes de grundlæggende byggeelementer, portkredsene, er opbygget og fungerer. Samtidigt har vi forsøgt at skrive bogen, så den kan læses med de forudsætninger, man kommer med fra forkeskolen.
Bogen er delt op i 5 kapitler. De 2 første omhandler portkredse - i kapitel 1 med dioder og transistorer og i kapitel 2 med de intergrerede kredse. I kapitel 3 omtales det binære talsystem og regning med binære tal, og i kapitlerne 4 og 5 omtales computerens opbygning. I kapitel 4 behandles de elektroniske regnelagre i computere og i kapitel 5 samles alle byggeelementerne til en computer.
For at gøre bogen fleksibel har vi delt hvert kapitel op i 3 dele: en hovedtekst, hvor begreberne introduceres; nogle øvelser, hvor eleverne afprøver tingene; og en supplerende tekst, hvor de interesserede har mulighed fore at gå i dybden, eller som kan læses af alle - evt. tværfagligt.
Det er ikke tanken, at man skal læse hele bogen eller lave alle øvelser. Tværtimod er det meningen, at man skal vælge emner og øvelser efter niveau, interesse og den afsatte tid. Selvom bogen starter med byggestenene til computere og slutter med den færdige computer, har det været hensigten, at det skulle være muligt at læse de enkelte kapitler som selvstændige emner.
Af konkrete eksempler på undervisningsforløb, hvor bogen har været anvendt, kan nævnes en 2.N, hvor følgende dele blev gennemarbejdet: Hovedtekst og udvalgte øvelser fra kapitel 1; Udvalgte øvelser fra kapitel 2; Hovedtekst og øvelser fra kapitlerne 3, 4 og 5 tværfagligt med matematik. Et andent forløb var et eksperimentelt speciale i en 3.F, hvor eleverne gruppevis aftalte indholdet med læreren.
Ved øvelser er det normalt et problem at skaffe tilstrækkeligt apparatur til, at en hel klasse kan lave de samme øvelser samtidigt. Men dels er det muligt at lade eleverne lave forskellige øvelser, og dels er komponenterne til øvelserne i de 3 første kapitler så billige, at det her er overkommeligt at anskaffe det nødvendige apparatur.
Øvelserne i kapitel 1 med dioder og transistorer laves lettest på "sømbrædder" (brug messingsøm), og ved øvelser med IC'ere har vi gode

erfaringer med prøveplader, hvor man kan forbinde komponenterne med prøveledninger. Sedler til at anbringe på prøvepladerne over benfordelingen for de meste benyttede IC'ere findes på side 111 og 112. De passer til prøvepladerne designet af Povl Vedelsby. Til øvelserne med komplicerede kredsløb er der for mange ben at styre på prøvepladerne, og derfor har vi udviklet nogle printplader: "RAM'en", "ALU'en" og "Computeren", hvor vi har gjort meget ud af at gøre dem så enkle og overskuelige, at eleverne har mulighed for at følge elektronikken.

Nødvendigt apparatur så som komponenter, prøveplader, prøveledninger og de større "brædder" kan købes hos Søren Fredriksen A/S.

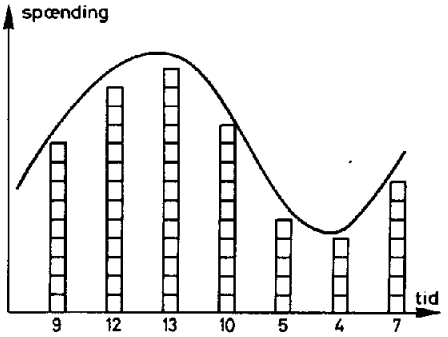
Vi takker alle, der har bidraget med forslag og konstruktiv kritik og håber, at denne bog må medvirke til at give digitalteknik og edb en naturlig placering i undervisningen.

Lisbeth Dragsted Niels Jørgen Vestergaard Niels Henrik Würtz

Indledning

Har du aldrig undret dig over, at alt nu skal være digitalt - ure, voltmetre, pladespillere, telefoner, fjernsyn mm.
Hvorfor sætter man mikroprocessorer i biler, vaskemaskiner, fotoapparater, hjul til fiskestænger og alverdens legetøj?
Og hvorfor er den digitale teknik i en så rivende udvikling netop nu?
Er det kun et modefænomen, eller er der fordele ved en digital teknik?
Denne bog er skrevet for at give en generel indføring i digitalteknik. Det er en fysikbog, men den skulle også gerne give svar på nogle af de generelle spørgsmål. Da bogen ikke måtte blive for omfattende, har vi valgt at se på digitalteknikken i mikroprocessorer og mikrodatamater, da det er her, digitalteknikken er gennemført mest ekstremt.
Mikroprocessoren er så fundamental, fordi den er generel. Den kan programmeres til næsten hvad som helst. Det samme kredsløb kan anvendes utroligt mange steder, fylder ingenting, og bliver masseproduceret meget billigt. F.eks. omtaler vi i denne bog mikroprocessoren, Z80, der er en chip med 40 ben, og som indeholder over 10.000 enheder. Den kan købes for ca. 30 kr i en elektronikforretning.

Analog eller digital?
Man inddeler den elektroniske teknik i to slags: analog- og digitalteknik. Vi vil forklare de to typer og skitsere, hvorfor digitalteknikken er den analoge overlegen.
Analogteknikken kendes blandt andet fra musikanlæg. En mikrofon omsætter lydsvingninger til elektriske signaler, hvor spændingens størrelse svarer til lydsvingningernes størrelse. Spændingen kaldes et analogt signal og er karakteriseret ved, at det er størrelsen af spændingen, der ræpræsenterer lydsvingningerne. I analogteknikken kan et signal forstærkes trinløst, og alle mellemværdier findes.
Digitalteknikken fungerer anderledes. I eksemplet fra før omsættes lydsvingningerne først til analoge spændinger. Disse elektriske spændinger måles digitalt nogle tusinde gange hvert sekund. Hver gang der måles, angives størrelsen af spændingen ved et helt tal målt i en eller anden spændingsenhed. Digitalis er latin og betyder "vedrørende fingrene". Spændingen deles op i "fingerbredder", og man får en talværdi. Lydsvingningerne er nu repræsenteret ved en masse tal (8000-40000 tal pr. sek).



Lagringen og transmissionen af disse tal foregår nu som lagring og transmission af almindelige tal. Vil man forstærke signalet, kan man blot multiplicere alle tallene med samme faktor i en elektronregnemaskine, før de atter omsættes til analoge spændinger og sendes til højttaleren.

1 Digitalisering af lyd.

Det kan måske undre, at kvaliteten af et musikanlæg bliver bedre ved, at signalerne digitaliseres. Lydsvingninger er analoge, og når de omsættes til digitale signaler, kan spændingssignaler mellem spændingsenhederne ikke gengives. Der opstår små fejl, som dog er uden betydning, når blot enhederne vælges tilstrækkeligt små. Man angiver spændingsstyrken med så mange cifre, at unøjagtigheden på sidste ciffer er underordnet. Gengivelsen bliver meget præcis, for man kan holde fuldstændigt styr på talværdierne. Intet tilføjes og intet fjernes.
Når det drejer sig om musikinstrumenter og musikgengivelse, er digitalteknikken dyrere og mere kompliceret end analogteknikken, men det kan hurtigt ændre sig, for komponenterne til digitalteknikken er lettere at masseproducere, og det er det samme fremstillingsudstyr, der benyttes til fremstilling af computere og andet digitalt udstyr. Priserne på analogt udstyr stiger meget voldsomt ved større nøjagtighed, hvorimod prisen for digitalt udstyr stiger jævnt med stigende nøjagtighed, idet man blot skal medtage lidt flere cifre og indbygge lidt mere apparatur af samme slags. Det er de samme grundlæggende elementer, der benyttes ved få og ved mange cifre.
For måleapparater, som f.eks. voltmetre og amperemetre, har den øgede efterspørgsel efter nøjagtige apparater bevirket en udvikling i retning af det digitale. For samme pris kan man købe digitale apparater, som er ca. 10 gange mere nøjagtige end tilsvarende analoge. Det er selvfølgelig en lettelse at slippe for at aflæse måleværdien på en skala, men det afgørende er, at selve måleprocessen foregår mere nøjagtigt.

God arbejdslyst.

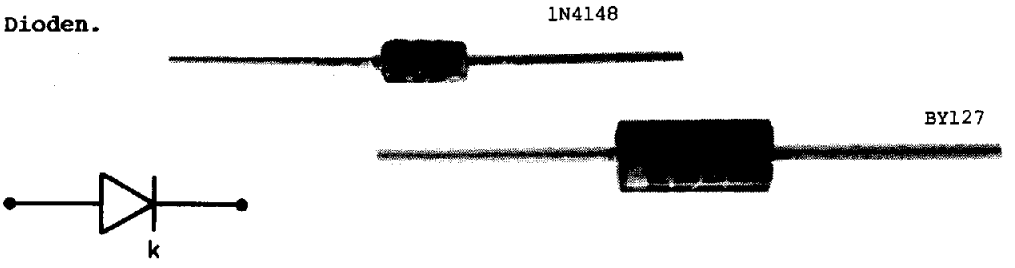
1. Dioder, Transistorer og gates

Oversigt

Hovedteksten i dette kapitel lægger vægt på at beskrive de egenskaber ved dioden og transistoren, som benyttes i digitalelektronikken. Her vises, hvorledes man kan opbygge portkredse, også kaldet gates, af dioder og transistorer. Ligeledes vises eksempler på, hvordan man sammensætter gates, så de kan udføre forskellige logiske operationer. Hovedtekstens teori afprøves eksperimentelt i øvelserne.
I det supplerende stof gives en mere generel beskrivelse af diodens og transistorens virkemåde ud fra en simpel halvlederteori. Desuden omtales de første regnemaskiner og den symbolske logik, der er den matematiske baggrund for digitalelektronikken.

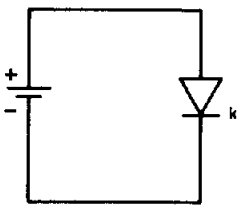
Hovedtekst

Dioden.



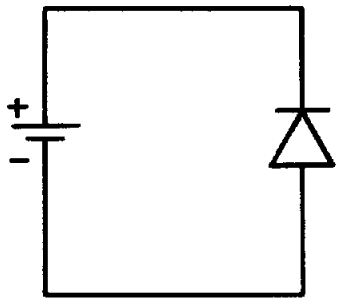
2 Symbol for diode.

Figur 2 viser symbolet for en diode sammen med eksempler på dioder. En diode har to ben (tilledninger), hvoraf den ene, katoden k, normalt er mærket med en prik eller en ring.
En diode kan forbindes til en jævnspændingskilde på to måder, som figur 3 og 4 viser. Forbindes dioden som vist på figur 3, går der en strøm i den retning, som pilen i symbolet angiver, når blot spændingen er over ca. 0,6 Volt. En diode kan altså lede strøm i pilens retning.



3 Diode i lederetningen.

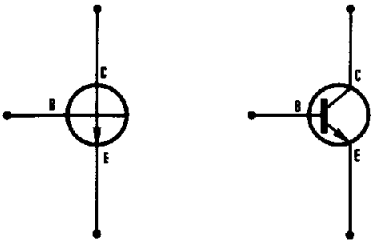
Når dioden er forbundet med spændingskilden som vist på figur 3, siger man, at dioden er forbundet i lederetningen. Bemærk, at katoden k er forbundet til minuspolen på spændingskilden.



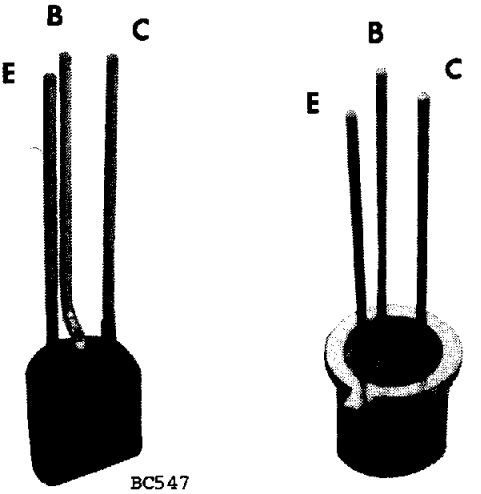
4 Diode i spærreretningen.

Vendes dioden, som vist på figur 4, går der ingen strøm, og man siger, at dioden er anbragt i spærreretningen. Ønskes mere information om opbygningen af en diode, henvises til det supplerende stof side 25.

Transistoren.



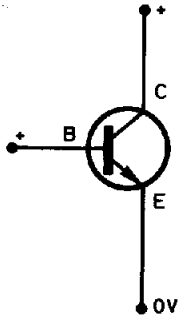
5 Symboler for n-p-n transistor.



BC547

BC107

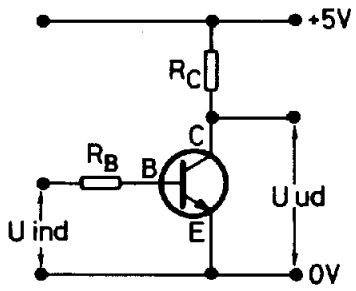
Figur 5 viser symboler for en transistor. Den har tre ben og kan være indbygget i et hus af plastik eller metal. I praksis kan den se ud som eksemplerne på figur 5, og den kan være konstrueret efter mange forskellige principper alt efter dens anvendelse. Vi vil dog kun omtale transistorer af typen n-p-n som f.eks. BC547.
Transistorens tre ben har navnene C for kollektor, B for basis og E for emitter.



6 Spænding til transistoren.

En n-p-n transistor forbindes normalt således, at basis og kollektor får en positiv spænding i forhold til emitteren. Der går derfor strøm gennem transistoren fra kollektoren og basis til emitteren. Se figur 6.
Kollektor-emitterstrømmen afhænger af basis-emitterstrømmen, og derfor kan man benytte basisstrømmen til at styre kollektor-emitterstrømmen.

Den spændingsstyrede transistor.



7 Spændingsstyret transistor.

I digitalteknikken anvendes en transistor aldrig alene. Den forbindes med modstande og styres af spændingen mellem basis og emitter, som figur 7 viser.
Emitteren forbindes til 0 V svarende til minus på batteriet, og alle andre spændinger måles i forhold hertil. Da basisstrømmen og dermed hele transistoren styres af basisspændingen (basis-emitterspændingen), kaldes denne for indgangsspændingen, U_{ind}. Kollektorspændingen kaldes tilsvarende U_{ud}.
Den spændingsstyrede transistor virker på den måde, at så længe indgangsspændingen er mindre end ca. 0,6 Volt, afbryder transistoren strømmen fra kollektor til emitter. Da transistorens modstand derved er mange gange større end R_c, vil næsten hele spændingsfaldet ligge over transistoren fra kollektor til emitter, og udgangsspændingen vil være lidt under de 5 V fra spændingsforsyningen. Øges indgangsspændingen, så den bliver større end 0,6 V, øges også kollektor-emitterstrømmen, og modstanden i transistoren falder. Når indgangsspændingen er over ca. 0,8 V, er modstanden i transistoren så lille, at vi kan betragte transistoren som kortsluttet mellem kollektor og emitter. Hele spændingsfaldet på 5 V ligger over modstanden R_c, og udgangsspændingen bliver tæt ved 0 V.

Når $U_{ind} < 0{,}6$ V, er U_{ud} ca. 5 V. Det vil sige Høj

Når $U_{ind} > 0{,}8$ V, er U_{ud} ca. 0 V. Det vil sige Lav

Transistoren kan altså styres, så udgangen kun får to niveauer, Høj og Lav. Transistoren fungerer digitalt.

Logiske kredsløb.

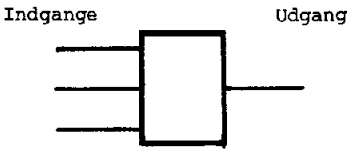
I det følgende vil vi beskrive en række logiske kredsløb, hvor spændingen Lav svarer til 0 V, og hvor spændingen Høj svarer til 5 V. I praksis anvendes intervallet 0 til ca. 0,6 V som Lav og intervallet 1 til 5 V som Høj. Vi benytter kun spændinger, som enten er høje eller lave, og kredsløbene bygges, så mellemværdier undgås.

Man kan, som det også vil fremgå af de følgende kapitler, opbygge logiske kredsløb, der kan udføre logiske operationer, tælle, regne, huske, styre etc.

Her kan Høj og Lav symbolisere:
et sandt eller falsk udsagn
et 1-tal eller et 0 i det binære talsystem
at en ordre skal udføres eller ej

De høje og lave spændingsværdier kan altså tolkes forskelligt afhængigt af, hvordan de logiske kredsløb anvendes. I praksis må tolkningen fremgå af sammenhængen.

Gates, Portkredse.



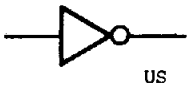
8 Symbol for GATES.

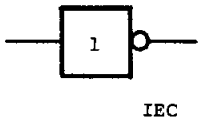
Byggestenene i de logiske kredsløb er de såkaldte gates eller portkredse. Disse har en eller flere indgange, men kun en udgang. Se figur 8.
En portkreds er karakteriseret ved, at der til enhver kombination på indgangene svarer én bestemt udgangsspænding (Høj = H eller Lav = L). Virkemåden beskrives ofte ved hjælp af en tabel.

NOT GATE, Inverter, Ikke-port.

A	$\overline{A}$
L	H
H	L

9 Inverteren.



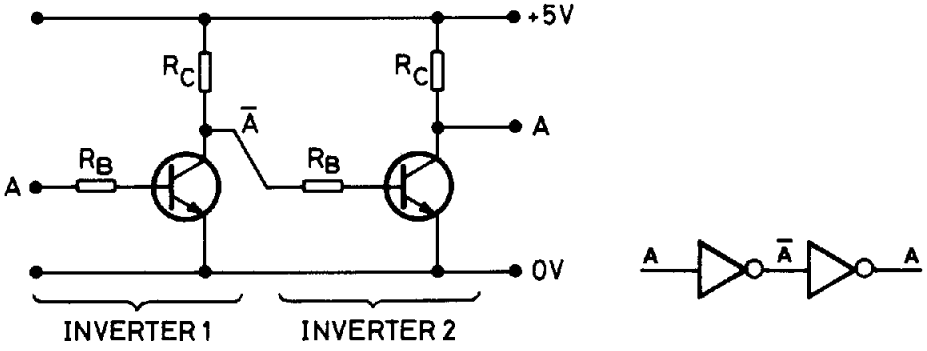


10 Symboler for inverteren

En NOT GATE, også kaldet en inverter eller en ikke-port, virker på den måde, at Lav spænding på indgangen giver Høj spænding på udgangen og omvendt. Dette er vist i tabellen på figur 9. Figur 10 viser symboler for en NOT GATE. Vi har valgt at benytte den amerikanske standard, fordi den er mere overskuelig: man kan lettere se, hvilken type gate der er tale om, og man kan lettere overskue, hvor indgangene og udgangene er placeret. I øvrigt angives denne standard altid i databøger og

benyttes i langt den største del af litteraturen. I dette kapitel vises desuden symbolerne efter den europæiske standard, IEC. Spændingerne angives ofte med symboler som f.eks. et A. En streg over symbolet betyder, at spændingen er modsat eller inverteret. Tolker man H som et sandt udsagn og L som et falsk udsagn, virker en NOT GATE som en logisk negation.
Af omtalen af den spændingsstyrede transistor side 11 fremgår det, at den virker som en NOT GATE, idet Lav på indgangen giver Høj på udgangen og omvendt.

Vi stiller det krav til kredsene, at udgangene skal kunne holde de lave og de høje spændinger helt adskilte. Men det er et ret strengt krav at stille, for når der løber strømme gennem modstande og dioder, vil der altid være spændingsfald over dem. Problemet løses lettest ved at anvende den spændingsstyrede transistor, der virker som en strøm- og spændingsforstærker. Som det også vil fremgå af øvelse 2, sender den spændingsstyrede transistor alle spændinger enten helt i top eller helt i bund. Derfor bliver alle gates, der skal anvendes i praksis, konstrueret med en spændingsstyret transistor i udgangen. Eventuelt anvendes en dobbeltinverter. En dobbeltinverter, også kaldet en buffer, består af to invertere efter hinanden. Udgangen giver samme spændingsværdier Høj eller Lav som indgangen, men i en meget "forbedret" udgave. På figur 11 ses kredsløbet for en sådan dobbeltinverter.



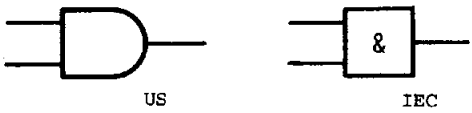
11 Dobbeltinverter og symbol for dobbeltinverter.

AND GATE, Og-port.

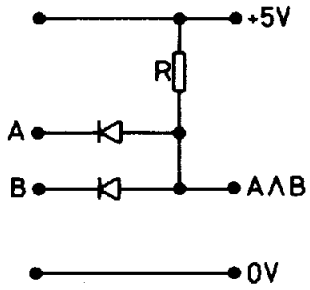
En AND GATE, også kaldet en og-port, virker på den måde, at udgangsspændingen er Lav, når blot en af indgangene er Lav. Kun når begge indgange er Høje, er udgangen Høj. Tabellen på figur 12 viser dens funktion og er samtidig tabellen for udsagnet A ∧ B, når Høj tolkes som sandt og Lav som falsk. Figur 13 viser symboler for en AND GATE.

A	B	A ∧ B
L	L	L
L	H	L
H	L	L
H	H	H

12 AND GATE.



13 Symboler for AND GATE.



14 AND GATE af dioder og modstande.

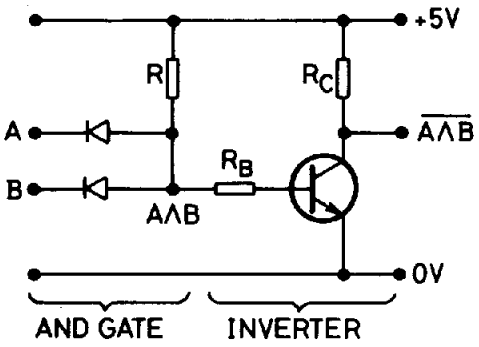
Figur 14 viser, hvorledes en AND GATE kan bygges af to dioder og en modstand. Er A eller B forbundet til 0 V, altså Lav, løber der en strøm gennem modstanden og dioden til 0 V. Spændingsfaldet over dioden er ca. 0,6 V, og resten af spændingsfaldet, ca. 4,4 V, falder over modstanden. Udgangen A ∧ B er Lav. Gøres begge indgange Høje, går der ingen strøm gennem modstanden, og der kommer derfor heller intet spændingsfald over den. Udgangen er Høj. Når denne kreds anvendes i praksis, efterfølges den af en dobbeltinverter.

NAND GATE, Ikke-og-port.

15 Symboler for NAND GATE.

Figur 15 viser symboler for en NAND GATE, også kaldet en ikke-og-port, der blot er en inverteret AND GATE. Bollen på symbolet angiver, at udgangen på AND GATE'n er inverteret. Figur 16 viser, hvorledes en NAND GATE kan opbygges.

Opgave 1. Opstil en funktionstabel for en NAND GATE.



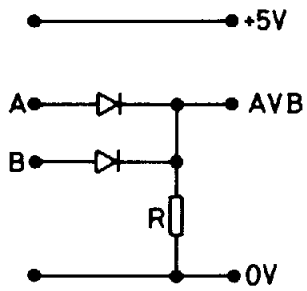
16 NAND GATE af AND GATE og Inverter.

OR GATE, Eller-port.

17 Symboler for OR GATE.

Figur 17 viser symboler for en OR GATE, også kaldet en eller-port, og figur 18 viser hvorledes en OR GATE kan opbygges af to dioder og en modstand. En OR GATE skal også i praksis efterfølges af en dobbelt-inverter.

A	B	A v B
L	L	L
L	H	H
H	L	H
H	H	H



18 Tabel over OR GATE og opbygning af OR GATE.

Opgave 2. Forklar, hvordan kredsløbet på figur 18 virker.

NOR GATE, Ikke-eller-port.

19 Symboler for NOR GATE.

Figur 19 viser symboler for en NOR GATE, også kaldet en ikke-eller-port, der blot er en inverteret OR GATE.

Opgave 3. Opstil funktionstabellen for en NOR GATE.

Opgave 4. Tegn et diagram over en NOR GATE opbygget af dioder, transistorer og modstande, og forklar, hvordan den virker.

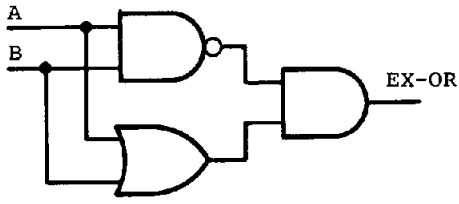
I det følgende omtales nogle eksempler på anvendelse af gates. De er valgt således, at der kun anvendes få kredse, så man selv kan bygge dem ved øvelserne til dette kapitel. Men eksemplerne skulle dog alligevel kunne give indtryk af, at princippet kan overføres til mere komplicerede kredse.

EXCLUSIVE-OR GATE.

En EXCLUSIVE-OR GATE er kun Høj på udgangen, når de to indgange er forskellige. Af tabellen på figur 20 fremgår, at en EX-OR GATE kan fremstilles af en NAND, en AND og en OR GATE. Se figur 21.

A	B	EX-OR
L	L	L
L	B	H
H	L	H
H	H	L

20 EXCLUSIVE-OR GATE.



21 EXCLUSIVE-OR af NAND, AND og OR.

Der er naturligvis udviklet teorier og teknikker til at sammensætte portkredse og gøre det så enkelt som muligt. Men da behovet for et grundigt kendskab til disse er aftaget væsentligt med fremkomsten af de integrerede kredse, hvor fabrikanterne har løst problemet for os, vil kun nogle af reglerne blive omtalt kort i det supplerende stof under symbolsk logik side 30ff.

Opgave 5. Vis ved hjælp af en sandhedstabel, at kredsløbet på figur 21 virker, som det skal. Lav en samlet sandhedstabel, der begynder med A, B, $\overline{A \wedge B}$, $A \vee B$.

Opgave 6. Vis som i opgave 5, at en EX-OR GATE også kan laves af $(A \wedge \overline{B}) \vee (\overline{A} \wedge B)$. Lav også et diagram svarende til det på figur 21.

Comparator.

En Comparator er en portkreds, der afgør, om to spændingsniveauer er ens eller ej. Den benyttes blandt andet i en datamat, når to bitmønstre skal sammenlignes.

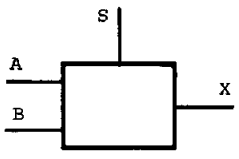
En Comparator laves ved at invertere udgangen fra en EX-OR GATE.

Opgave 7. Lav en funktionstabel for Comparatoren.

Opgave 8. Vis, at en Comparator kan laves som $(A \wedge B) \vee \overline{(A \vee B)}$. Tegn også et kredsløbsdiagram med symboler.

Multiplexer.

En Multiplexer er et kredsløb med to indgange A og B , en styreindgang S og en udgang X. Med styreindgangen S vælger man, om A eller B skal overføres til udgangen X.

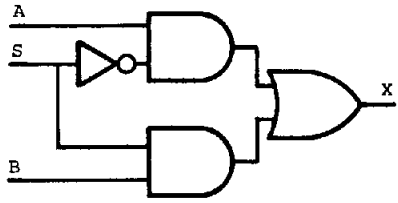


X = A for S = L

X = B for S = H

S	A	B	X
L	L	L	L
L	L	H	L
L	H	L	H
L	H	H	H
H	L	L	L
H	L	H	H
H	H	L	L
H	H	H	H

22 Symbol og tabel over Multiplexer.

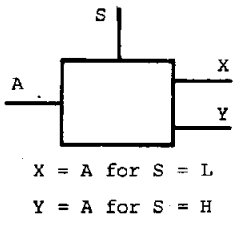


23 Multiplexer.

Opgave 9. Vis, at kredsløbet på figur 23 giver det ønskede.

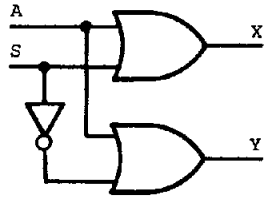
Multiplexeren benyttes, når mange signaler skal transporteres på den samme ledning, f.eks. ved digital telefoni. Her sendes på skift signaler fra 64 abonnenter på den samme ledning. For at dette kan lade sig gøre, deles signalet fra den enkelte abonnent op og komprimeres, så det kan sendes på kortere tid. I den anden ende af ledningen skal der så være et tilsvarende kredsløb, der kan adskille signalerne fra de 64 abonnenter igen. Et sådant kredsløb kaldes en demultiplexer.

Demultiplexer.
En Demultiplexer virker modsat multiplexeren. Den har kun en egentlig indgang A, en styreindgang S og to udgange X og Y. Se figur 24. Med styreindgangen vælger man, om indgangssignalet skal til udgangen X eller Y.



S	A	X	Y
L	L	L	H
L	H	H	H
H	L	H	L
H	H	H	H

24 Symbol og tabel over Demultiplexer.



25 Demultiplexer.

Figur 25 viser, hvorledes kredsløbet kan fremstilles af to OR GATES og en NOT GATE.

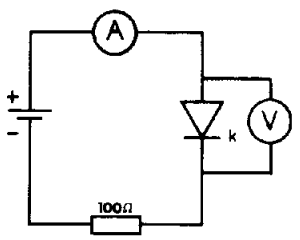
Opgave 10. Vis, at kredsløbet fungerer som en demultiplexer.

Øvelse 1. Måling af diodens spændingskarakteristik.

For en modstand gælder, at spændingsfaldet U over den og strømstyrken I gennem den er proportionale, hvilket udtrykkes i Ohms lov $U = R \cdot I$. En modstand har en bestemt resistans R. Men dette gælder ikke for dioder, som vi skal se i denne øvelse.

Til øvelserne anvendes en variabel spændingskilde, et amperemeter, et voltmeter, en beskyttelsesmodstand og en diode. Apparaturet samles som vist på figur 26, og det gøres lettest på følgende måde:

1) Skru ned for spændingen til 0 V.
2) Forbind først den strømførende kreds ved f.eks. at starte ved + på spændingskilden og følge strømmen i dens retning indtil spændingskildens negative pol. Strømmen i amperemetret skal altid løbe ind ved + og ud ved -. Amperemetret skal være indstillet på et stort måleområde.
3) Til slut forbindes voltmetret således, at den lille strøm, der går gennem det, kommer ind ved + og ud ved -.
4) Når der skrues op for spændingen, følges visningen på metrene, og de stilles på områder, så visningen bliver størst mulig.



26 Diode i lederetningen.

Mål sammenhørende værdier af spændingen over dioden og strømstyrken gennem den. Vær særlig grundig hvor dioden begynder at lede strømmen, hvilket sker ved 0,6-0,8 V for siliciumdioder og ved 1,6-2,0 V for lysdioder.

Modstanden, der er anbragt i serie med dioden, er medtaget, for at man ikke ved en fejltagelse skal komme til at brænde dioden af. Små dioder kan klare ca. 50 mA og store omkring 1 A. Lysdioder kan kun tåle 20-30 mA.
Vend dioden og lav igen en serie målinger.

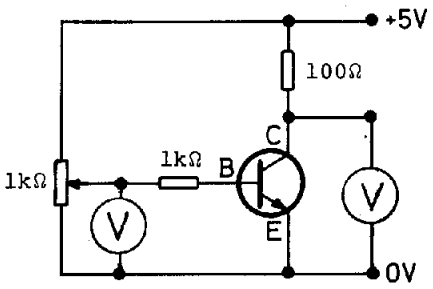
Ud fra måleresultaterne laves en karakteristik af dioden, hvilket gøres ved at tegne målepunkterne ind i et koordinatsystem, hvor spændingsfaldet afsættes ud ad x-aksen og strømstyrken op ad y-aksen. Tegn en glat kurve gennem målepunkterne.

Beskriv diodens egenskaber i ord.

Øvelse 2. Karakteristik af den spændingsstyrede transistor.

I denne øvelse skal vi se, hvordan transistoren fungerer i digitalteknikken.

Transistoren skal være af typen n-p-n og kan f.eks. være en BC547 eller en BC172.
Lav opstillingen vist på figur 27. Det er nok en god ide at vente med de to voltmetre til sidst. Transistorernes benfordeling er for de to nævnte transistorer vist på figur 5 side 10.



27 Spændingsstyret transistor.

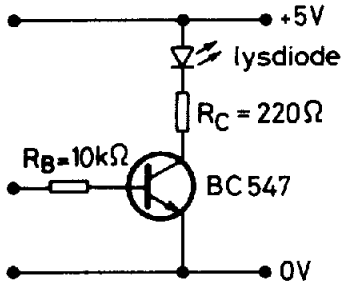
Indgangsspændingen reguleres mellem 0 og 5 V ved hjælp af skydemodstanden, der f.eks. kan være på 1 kΩ. Mål sammenhørende værdier mellem indgangsspændingen og udgangsspændingen. Foretag særligt mange målinger, hvor udgangsspændingen ændrer sig meget.

Indtegn målepunkterne i et koordinatsystem med indgangsspændingen afsat ud ad x-aksen og udgangsspændingen op ad y-aksen. Tegn en glat kurve gennem målepunkterne. Hvad viser denne?

Find evt. ud af, hvor meget basismodstandens størrelse betyder for transistoren anvendt som inverter. Gentag forsøget med en anden basismodstand.

Prøv til sidst at koble to spændingsstyrede transistorer efter hinanden til en dobbeltinverter. Opstillingen er vist på figur 11 side 13. Forklar ud fra grafen, hvordan dobbeltinverteren kan anvendes i digitalteknikken.

Udlæsemodul.

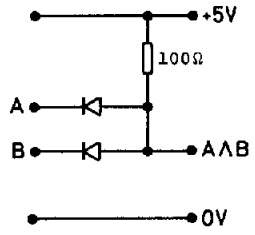


28 Udlæsemodul.

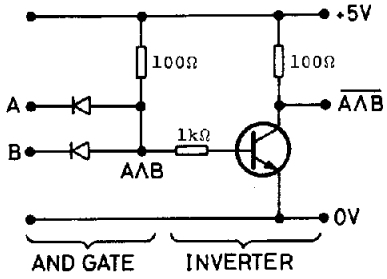
I mange øvelser i denne bog vil vi teste, om en spænding er Høj eller Lav, ved hjælp af en lysdiode i et kredsløb, som vist på figur 28. Bemærk, at kredsløbet blot er den spændingsstyrede transistor med lysdioden monteret i serie med kollektormodstanden. Dioden lyser, når transistoren begynder at lede strøm, hvilket sker ved indgangsspændinger over ca. 0,8 V.

Som de efterfølgende forsøg gerne skulle vise, virker portkredsene opbygget af dioder og transistorer efter hensigten, når vi vedtager, at spændingen Høj får lysdioden til at lyse, og at spændingen Lav ikke gør det. Vi forsøger at konstruere portkredsene så udgangsspændingerne ikke falder i spændingsområdet 0,6-0,8 V.

Øvelse 3. Test af AND GATE og NAND GATE med udlæsemodul.



29 AND GATE.

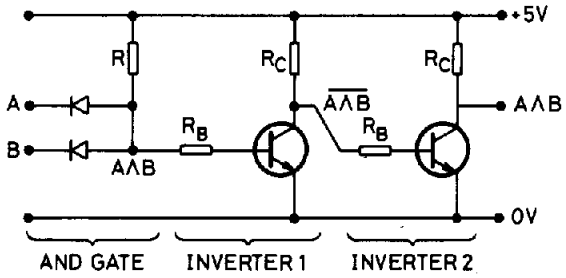


30 NAND GATE.

Ovennævnte portkredse kan f.eks. være opbygget på sømbrædder.
Hvis man bygger AND GATE'n af lysdioder, skal man af hensyn til spændingsfaldet på ca. 1,8 V over dioden anbringe en ekstra lysdiode i udgangen med katoden mod udgangen.

Test udgangsspændingerne på de to portkredse, når indgangene har alle fire kombinationer af 0 og 5 V. Virker de som forventet?
Forbind også to AND GATES i serie således, at udgangen på den første AND GATE forbindes til den ene af indgangene på den næste. Forbind den ene af indgangene i den første portkreds til 0 V og undersøg, om udgangen på den sidste bliver Høj eller Lav. Hvorfor er resultatet ikke som forventet?

Man har ikke tilsvarende problemer med NAND GATES. En NAND GATE kan godt trække så meget strøm, at indgangen på en efterfølgende gate bliver Lav, når den skal være det, Her udnyttes transistorens strømforstærkende egenskaber.



31 AND GATE med dobbeltinverter.

Konstrueres en AND GATE, som figur 31 viser, med en dobbeltinverter, vil denne også virke efter hensigten, når flere portkredse sættes sammen, hvilket flere af de efterfølgende øvelser gerne skulle vise. Test, om to AND GATES virker korrekt, når de sættes i serie.

Den simple OR GATE, omtalt i hovedteksten (figur 18), skal også efterfølges af en dobbeltinverteret port for at virke i praksis. Efterprøv evt. dette.

En række portkredse, med ensartet opbygning, som virker efter hensigten, når de sættes sammen, kaldes en logisk familie.
Vi har set en sådan logisk familie af AND, NAND, OR og NOR GATES, der alle har en spændingsstyret transistor i udgangen. I kapitel 2 anvendes den integrerede logiske familie, TTL, det vil sige Transistor Transistor Logic kredse.

Øvelse 4. Test af sammensatte portkredse.

Opbyg EXCLUSIVE-OR GATE, Comparator, Multiplexer og Demultiplexer og undersøg, om de virker efter hensigten.
Både AND og OR GATE skal have en dobbeltinverter i udgangen. Øvelsen er naturligvis også en test af, om de anvendte portkredse udgør en logisk familie.

Øvelse 5. De Morgans regler.

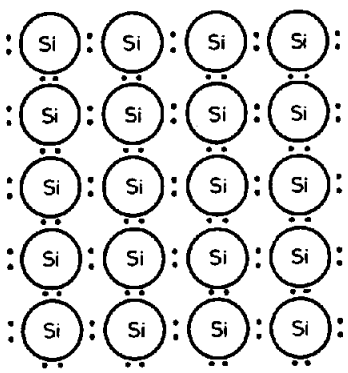
Afprøv rigtigheden af De Morgans regler omtalt i det supplerende stof side 34 ø. Vis f.eks. $\overline{A \wedge B} = \overline{A} \vee \overline{B}$.

Supplerende stof

Halvledere.

I det følgende gives en kort beskrivelse af halvledere, specielt dioders og transistorers virkemåde.
For at forstå opbygningen af halvledere vil vi først se på opbygningen af metaller med kobber som eksempel. Kobber er placeret i første sidegruppe i det periodiske system, og med 29 elektroner er K, L og M skallerne fyldt helt op med henholdsvis 2, 8 og 18 elektroner, mens den sidste elektron er placeret i N skallen. Da kobber er et metal, er atomerne placeret i et rumligt krystalgitter, hvor den enlige elektron i N skallen binder atomerne sammen, men den er så løst bundet til sit eget atom, at den er frit bevægelig i hele krystallen. Elektronerne i de fyldte skaller er stærkt bundet til det enkelte atom og bidrager ikke væsentligt til bindingerne mellem atomerne. Når kobber kan lede strøm, skyldes det den frie elektron, der kan bevæge sig gennem hele krystallen og dermed let kan transportere ladning. Da der er mange frie elektroner i kobber (en pr. atom), er kobber en god leder.

I digitalteknikken anvendes silicium = Si som halvledermateriale. I grundstoffernes periodiske system er det placeret i fjerde hovedgruppe sammen med kulstof (carbon), germanium og tin. Disse fire stoffer har alle 4 elektroner i yderste elektronskal og kan derfor danne bindinger til 4 naboatomer ved hjælp af fælles elektronpar i en såkaldt diamantgitterstruktur. Denne har naturligvis en rumlig udstrækning, men er vist plant på figur 34. De to prikker mellem to atomer skal symbolisere de to fælles elektroner, der binder atomerne sammen.



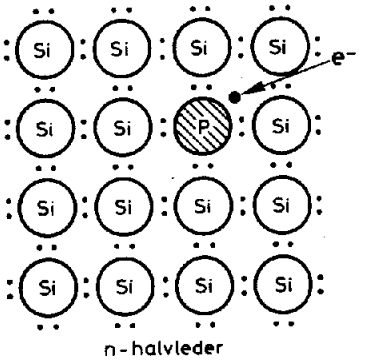
Siliciumkrystal

34 Siliciumkrystal.

Jo mindre atomet er, jo vanskeligere har elektronerne ved at slippe løs. Diamant er således en isolator, da bindings-elektronerne mellem kulstofatomerne er så kraftigt bundet, at der ingen frie elektroner er til at lede den elektriske strøm. Derimod er Si en halvleder, fordi der allerede ved stuetemperatur slås enkelte elektroner løs af krystalgitteret. Men det er meget få, ca. 1 for hver 10^{13} atomer. Navnet halvleder hentyder derfor til, at Si kan lede strøm ganske lidt.

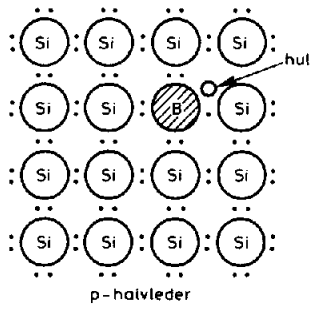
I germanium er der flere løstsiddende elektroner, og i tin er der endnu flere. Tin har metalkarakter med mange frie elektroner.
Når en elektron slås ud af sin position i krystalgitteret i silicium, kan den lede strøm ligesom i et metal. Nu mangler der en elektron, og vi siger, at der opstår et hul i gitterstrukturen. Dette hul kan imidlertid flytte sig indirekte ved, at en elektron fra et naboatom flytter over i det, men så opstår der et nyt hul, hvor elektronen kom fra, etc.
De frie elektroner bevæger sig modsat strømmens retning, da de jo har en negativ ladning. De er ikke indbygget i krystallens regelmæssige struktur og kan bevæge sig helt frit. Huller bevæger sig i strømmens retning og er bundet til at bevæge sig fra atom til atom.
Halvledere anvendes ikke rene. Ledningsevnen forøges væsentligt ved såkaldt doping, hvor man fordamper atomer fra andre hovedgrupper i det periodiske system ind i den rene Si-krystal. Herved opstår n- og p-halvledere.

n-halvleder.
Dopes Si-krystallen med atomer fra femte hovedgruppe (med 5 elektroner i yderste elektronskal - f.eks. phosphoratomer) således, at disse indbygges i krystalstrukturen, vil der være en elektron i overskud for hvert phosphoratom. Se figur 35. Den femte elektron er der ikke plads til i gitterstrukturen, og den vil kunne bevæge sig frit gennem krystallen. Der kommer en fri elektron pr. fremmedatom. Ledningsevnen kan således forøges mange gange ved passende doping. Da ledningsevnen hovedsageligt skyldes de negative ladninger (elektronerne), kaldes krystallen også en n-halvleder. n-halvlederen er elektrisk neutral, da der findes lige så mange positive ladninger, som der er elektroner. De frie elektroner hidrørende fra forureningsatomerne kan bevæge sig frit, mens de tilsvarende positive ladninger sidder fast i gitteret i kernerne på forureningsatomerne.



35 n-halvleder.

p-halvledere.



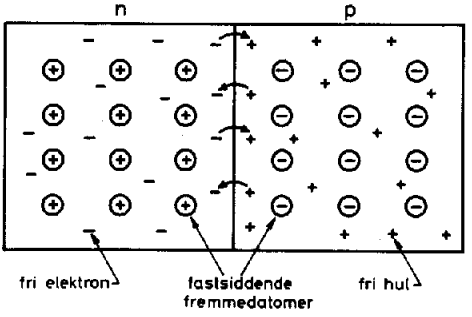
36 p-halvleder.

Dopes Si-krystaller med atomer fra tredie hovedgruppe i det periodiske system (f.eks. med boratomer) således, at disse atomer indbygges i krystalstrukturen, vil der være en elektron i underskud - et hul - ved hvert boratom. Se figur 36. Disse huller kan let fyldes af elektroner fra naboatomerne, og hullerne kan dermed flytte sig. Ledningsevnen skyldes her de positive ladningsbærere (huller), og krystallen kaldes en p-leder. En p-leder er også elektrisk neutral, men ledningsevnen er forøget væsentligt.

Man doper normalt af størrelsesorden 1 fremmedatom for hver 10^4-10^7 Si-atomer. Det er så mange, at vi kan tillade os at se bort fra antallet af elektron-hulpar i den rene halvleder og betragte en n-halvleder som en leder, hvor strømmen udelukkende ledes af elektroner, og en p-halvleder som en leder, hvor strømmen udelukkende ledes af huller. Først, når man fremstiller Si-krystaller med både n- og p-halvledere, får man de komponenter, dioder og transistorer, der har så mange praktiske anvendelser.

Dioden.

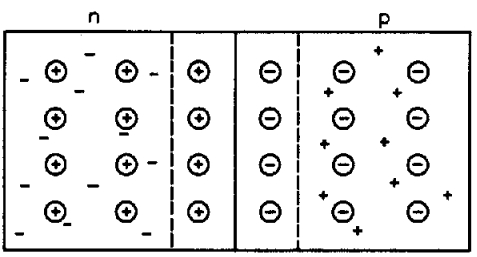
En diode består af en p-halvleder i kontakt med en n-halvleder.



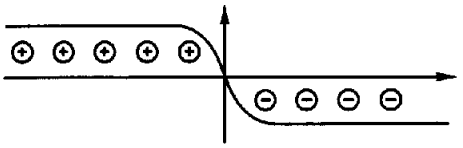
37 Dioden af n- og p-materiale.

I p-halvlederen flytter hullerne sig tilfældigt rundt på grund af termisk bevægelse. Tilsvarende bevæger de frie elektroner sig tilfældigt rundt i n-halvlederen. I grænselaget mellem de to typer halvledere bevæger elektroner fra n-halvlederen sig ind i p-halvlederen, hvor de udfylder huller. Tilsvarende bevæger hullerne i p-halvlederen sig ind i n-halvlederen, hvor de udfyldes af elektroner. n-halvlederens grænselag er nu ikke længere neutralt. De fastsiddende positive kerner fra femte hovedgruppe i n-laget mangler en elektron. Der opstår et rumladningsområde, der er positivt, bestående af fastsiddende kerner. På tilsvarende måde opstår et negativt

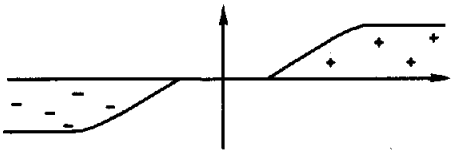
område i p-laget bestående af de fastsiddendede atomer fra tredie hovedgruppe med en elektron ekstra. Figur 37 viser, hvordan der opstår et rumladningsområde i grænselaget ved diffusion af huller og elektroner. Denne proces modvirkes efterhånden af de fastsiddende ladninger i grænseområdet. Elektronerne fra n-halvlederen bliver frastødt af de negative rumladninger i p-halvlederen. Et hul i p-halvlederen frastødes af positiv rumladning i n-halvlederen. Jo flere rumladninger, jo større frastødning. Der opstår en ligevægtstilstand, hvor disse frastødningskræfter forhindrer yderligere diffusion. Figur 38 viser denne ligevægtstilstand med et rumladningsområde, der er tegnet ret bredt. I virkeligheden er det meget tyndt, ca. 10^{-4} mm. Bredden øges lidt med øget temperatur. På figuren er alle ladninger, der er angivet med en ring, fastsiddende ladninger på forureningsatomerne, som derfor ikke kan flytte sig. Figur 39 viser, hvorledes antallet af ladninger fordeler sig i dioden.



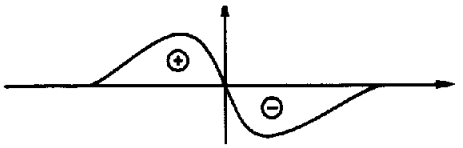
38 Grænseområdet i dioden uden frie ladningsbærere.



39 Fordeling af ladninger i dioden.
a) Faste ladninger på forureningsatomerne.



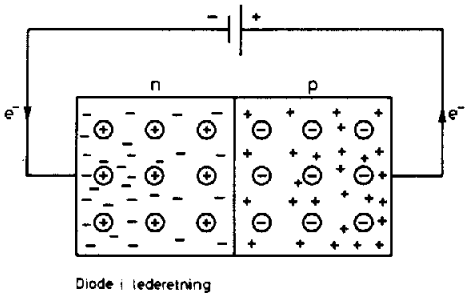
b) Frie ladningsbærere (elektroner og huller).



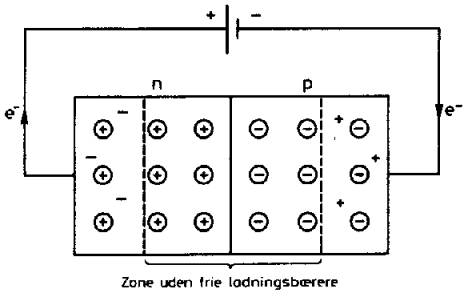
c) Samlet ladning.

En diode kan forbindes til en ydre spændingskilde på to måder. Forbindes n-halvlederen til spændingskildens negative pol og p-halvlederen til den positive pol som vist på figur 40, vil spændingskilden pumpe elektroner ind i n-halvlederen og trække elektroner ud af - dvs. pumpe huller ind i - p-halvlederen. Er spændingsfaldet over dioden større end 0,6 V (for siliciumdioder), nedbrydes rumladningsområdet fuldstændigt.
Der kommer et overskud af frit bevægelige elektroner i n-halvlederen, som forskydes mod p-laget, og et overskud af frit bevægelige huller i p-halvlederen, der forskydes mod n-laget (se figur 41). Elektroner fra n-halvlederen kan let passere ind i p-halvlederen, og huller fra p-

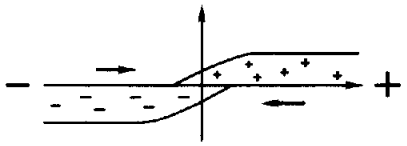
halvlederen kan let passere den modsatte vej. Spændingskilden driver derved elektroner gennem dioden i retning fra n- til p-halvlederen og huller den modsatte vej. Dioden leder elektrisk strøm. Ved lysdioder udsendes lys, når de frie elektroner i grænseområdet rekombinerer med et hul. Rødt lys svarer til en spændingsforskel på 1,8 V, og derfor leder røde lysdioder først strøm ved en spænding på ca. 1,8 V.



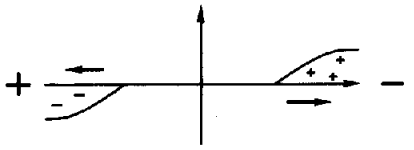
40 Diode i lederetningen.



42 Diode i spærreretningen.



41 Fordeling af frie ladninger.



43 Fordeling af frie ladningsbærere.

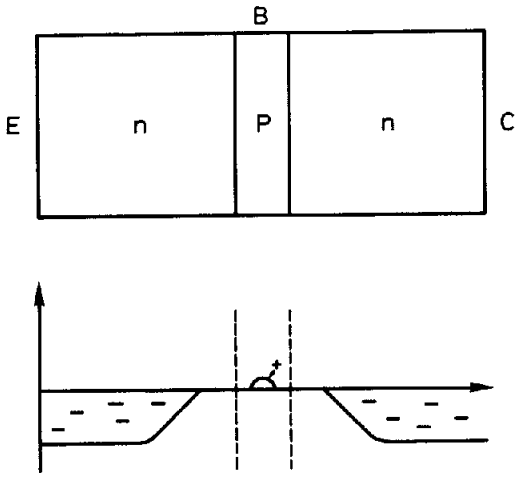
Diode i spærreretningen.

Forbindes spændingskildens positive pol til n-halvlederen og den negative pol til p-halvlederen (se figur 42), vil spændingskilden trække elektroner ud af n-halvlederen og pumpe elektroner ind i - og dermed trække huller ud af - p-halvlederen. Grænseområdet, hvor der ingen bevægelige ladningsbærere er, bliver derved endnu større, og der kan ikke gå nogen strøm. Dioden spærrer for strøm.

Denne usymmetri i dioden kan anvendes til ensretning af strømme.

Den bipolære transistor.

I det følgende omtales kun den såkaldte n-p-n transistor, hvor en siliciumkrystal er dopet således, at en p-halvleder er placeret mellem to n-halvledere som vist på figur 44. Beskrivelsen af en p-n-p transistor bliver magen til, blot byttes om på ordene elektron og hul, og spændinger og ladningssymboler skifter fortegn.

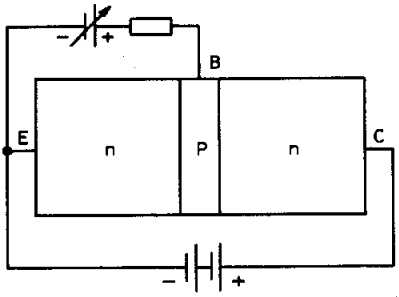


44 Transistor af n-, p- og n-materiale.

Man kan opfatte n-p-n transistoren som to modsat rettede dioder med fælles p-lag. I de to grænselag dannes rumladninger ved termisk diffusion, som beskrevet i afsnittet om dioder. Figur 44 viser fordelingen af bevægelige ladningsbærere i transistoren. Det midterste p-lag er meget tyndt (ca. 10^{-6} m) og kaldes transistorens basis. De to andre kaldes kollektor og emitter. Emitteren dopes sædvanligvis kraftigere end basis.

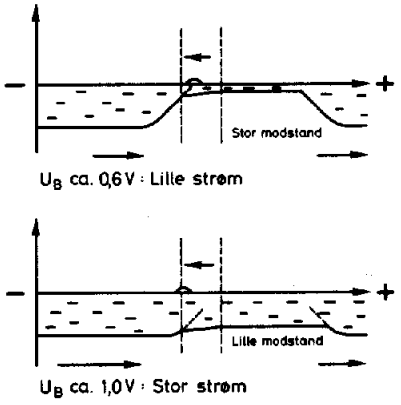
Transistorens forspændinger.

Når en n-p-n transistor forbindes til en spændingskilde, tilføres både basis og kollektor en positiv spænding i forhold til emitteren.



45 n-p-n transistorens forspænding.

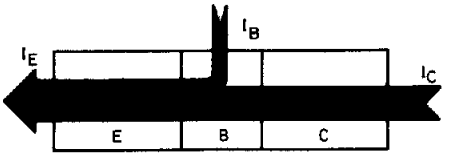
Emitteren bliver det fælles nulpunkt for både basis-emitterspænding, U_{BE}, og kollektor-emitterspænding, U_{CE}. Se figur 45. Man sørger for, at $U_{BE} < U_{CE}$. Herved bliver basis-emitter en diode forspændt i lederetningen og kollektor-basis en diode forspændt i spærreretningen. Den tidligere beskrivelse af en diode i lederetning og spærreretning kan nu direkte overføres til beskrivelse af transistoren.



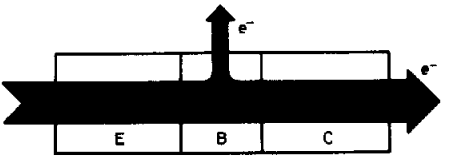
46 Fordeling af frie ladningsbærere

Da basis-emitter dioden er forspændt i lederetningen, forskydes fordelingen af de negative ladningsbærere (de frie elektroner i emitterlaget) mod basislaget, og er basisspændingen over ca. 0,6 V, kan de frie elektroner fra emitterlaget nå ind i basislaget og give anledning til en strøm gennem basis.

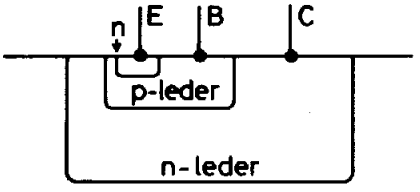
Men så snart der er negative ladningsbærere (elektroner) i basis-p-laget, hvor der normalt kun er positive ladningsbærere (huller), vil elektronerne tiltrækkes af kollektorens positive spænding. Transistoren leder nu strøm fra kollektor til emitter.
Med basisspændingen bestemmer vi antallet af frie elektroner i basislaget, og da det er disse elektroner, der bevæger sig fra emitteren til kollektoren, styres kollektor-emitterstrømmen af basisspændingen. Modstanden fra emitter til kollektor er også bestemt af antallet af ladningsbærere i basislaget, og som før afhænger antallet af basisspændingen, så derfor falder modstanden i transistoren med stigende basisspænding.
Kun en lille brøkdel (1/1000-1/100) af de frie elektroner fra emitter-n-laget opfanges af huller i det tynde basislag, hvor de atter trækkes ud af basis-emitter spændingskildens positive pol. Når transistoren leder strøm, vil der derfor altid gå en lille strøm gennem basisledningen. Men tabet af frie elektroner sker kun i basisområdet, for når elektronerne først er nået ind i kollektorens n-lag, er der ingen huller, og elektronerne kan bevæge sig uhindret mod kollektorens positive pol. Figur 47 viser elektronbevægelsen gennem n-p-n transistoren med forspændinger som på figur 45.
Desuden viser figur 48 kollektorstrømmen, basisstrømmen og emitterstrømmen gennem n-p-n transistoren. Strømmen er jo modsat den negative elektronbevægelse.



47 Elektronbevægelse gennem en transistor.



48 Strøm gennem en transistor.



49 Opbygning af transistor i en krystal.

I praksis er n-p-n transistoren fremstillet af en n-halvlederkrystal, hvor der er indføjet nogle områder af p- og af n-halvledermateriale. Kollektor, basis og emitter forbindes som vist på figur 49. For at gøre basisstrømmen mindst mulig, skal basis-p-laget være meget tyndt.

Boolsk algebra - logikkens love.

Portkredsene, der netop kan have to adskilte tilstande: Høj eller Lav, kaldes også logiske kredsløb, fordi det viser sig, at de regler, der gælder for deres sammensætning, er de samme, som gælder for logiske udsagn. Disse kan jo netop have to sandhedsværdier: Sand eller falsk. Grundlaget for en logikteori blev lagt allerede af den græske filosof Aristoteles ca. 350 f.Kr., men først i 1847 formulerede englænderen George Boole i sin bog "An Investigation of the Law of Thought" en matematisk beskrivelse af de regler, der gælder for den symbolske logik. Boole beskriver sine mål således: "Hensigten med den følgende afhandling er at undersøge de fundamentale love for de tankeopperationer, ved hvis hjælp vor ræsonneren udføres, at give dem udtryk i en kalkule og på dette grundlag opbygge logikken og konstruere dens metode."
I det følgende kan vi kun give en kortfattet omtale af disse love, formuleret i det sprog, der er kendt fra matematikundervisningen, og som er let at overføre til den logiske kredsløbsteori.
Boole indførte bogstaver til at symbolisere udsagn (f.eks. A: Det sner, og B: Det regner). Bogstaverne kan også symbolisere ting og begreber, så Booles teori er mere almen, end den fremtræder i det følgende. Man kan så forbinde bogstaverne med tegn og derved danne sammensatte udsagn. Det snilde er så, at man kan opstille regler for sammensætning af udsagn, som er uafhængig af, hvad bogstaverne står for. Og det bliver langt lettere at finde sandhedsværdier af mere komplicerede udsagn, når de angives på en symbolsk form.

Sammensatte udsagn.

Af de to udsagn A og B kan dannes det sammensatte udsagn "A og B", der regnes for sandt, når begge udsagn er sande, og falsk ellers. Vi anvender symbolet " $\wedge$ " for "og". Sandhedsværdien for det sammensatte udsagn afhænger af såvel A som B's sandhedsværdier og kan f.eks. angives ved hjælp af en tabel, hvor alle muligheder er medtaget (se figur 50). På tilsvarende måde dannes udsagnet "A eller B", der betegnes ved A v B. Det negerede udsagn angives ved en streg over bogstavsymbolet: $\overline{A}$ = "ikke A".

A	B	A $\wedge$ B
f	f	f
f	s	f
s	f	f
s	s	s

50 A∧B

A	B	C	$(A \wedge B) \wedge C$	$(A \vee B) \vee C$
f	f	f	f	f
f	f	s	f	s
f	s	f	f	s
f	s	s	f	s
s	f	f	f	s
s	f	s	f	s
s	s	f	f	s
s	s	s	s	s

51 Sandhedstabel for $(A \wedge B) \wedge C$ og $(A \vee B) \vee C$.

Man kan også sammensætte flere udsagn. Med 3 symboler kan man f.eks. danne $(A \wedge B) \wedge C$ og $(A \wedge B) \vee C$. Se sandhedstabellen på figur 51, der her må få 8 linier - overvej hvorfor.

Et fundamentalt tegn i den symbolske logik er identitetstegnet, der sættes mellem udsagn, der har samme sandhedsværdier dvs. samme sandhedstabel. I det følgende benyttes lighedstegnet som symbol på, at to udsagn er identiske.

Opgave 11. Opstil sandhedstabeller for negationen og $\vee$.

Vi vil omtale nogle vigtige eksempler på sammensatte udsagn.

Implikationen.

$A \Rightarrow B$ (A medfører B), der har den viste sandhedstabel (figur 52), anvendes, når man drager logiske slutninger f.eks. i matematiske beviser. $A \Rightarrow B$ er kun falsk, når A er sand, og B er falsk. $\Rightarrow$ kaldes ofte medførerpilen og oversættes til "Hvis A er sand, er B også sand". Hvis A er sand og B falsk, er det sammensatte udsagn naturligvis falsk. Derimod er det måske mindre indlysende, at man tillægger $A \Rightarrow B$ sandhedsværdien sand, når A er falsk. Men det bringer implikationen til at være identisk med udsagnet: "Det er ikke sandt, at både A er sandt, og B er falsk". Og det lyder jo rimeligt?

A	B	$A \Rightarrow B$
f	f	s
f	s	s
s	f	f
s	s	s

52 $A \Rightarrow B$

A	B	$\overline{B}$	$A \wedge \overline{B}$	$\overline{A \wedge \overline{B}}$
f	f	s	f	s
f	s	f	f	s
s	f	s	s	f
s	s	f	f	s

53 $A \Rightarrow B = \overline{A \wedge \overline{B}}$

Opgave 12. Vis ved hjælp af en sandhedstabel, at $A \Rightarrow B = \overline{A} \vee B$.

Biimplikationen.

$A \Leftrightarrow B$ (A er ensbetydende med B) har den på figur 54 viste sandhedstabel. Symbolet anvendes, når man drager logiske slutninger, der kan gå begge veje, f.eks. når man fører matematiske beviser.

$A \Leftrightarrow B$ er sand, når A og B har samme sandhedsværdi, og falsk ellers. Dette er i overensstemmelse med sandhedstabellerne på figur 55.

Matematiske beviser føres som regel med udgangspunkt i nogle forudsætninger (definitioner og antagelser). Og man viser da, at disse medfører den egenskab, man undersøger. Man viser, at forudsætninger $\Rightarrow$ egenskab. Forudsætningerne er tilstrækkelige. Hvis man også kan vise, at forudsætningerne følger af egenskaben, har man vist sætningen: Egenskab $\Leftrightarrow$ forudsætning. Denne bevismetode udnytter, at $A \Leftrightarrow B = (A \Rightarrow B) \wedge (A \Leftarrow B)$.

A	B	$A \Leftrightarrow B$
f	f	s
f	s	f
s	f	f
s	s	s

54 $A \Leftrightarrow B$

A	B	$\overline{A}$	$\overline{B}$	$\overline{A} \wedge \overline{B}$	$A \wedge B$	$(\overline{A} \wedge \overline{B}) \vee (A \wedge B)$
f	f	s	s	s	f	s
f	s	s	f	f	f	f
s	f	f	s	f	f	f
s	s	f	f	f	s	s

55 $A \Leftrightarrow B = (\overline{A} \wedge \overline{B}) \vee (A \wedge B)$

Opgave 13. Vis, at $A \Leftrightarrow B = (A \Rightarrow B) \wedge (A \Leftarrow B)$.

Der kan dannes $2^4 = 16$ forskellige udsagn ud fra to udsagn A og B:

A	B	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}
f	f	f	f	f	f	f	f	f	f	s	s	s	s	s	s	s	s
f	s	f	f	f	f	s	s	s	s	f	f	f	f	s	s	s	s
s	f	f	f	s	s	f	f	s	s	f	f	s	s	f	f	s	s
s	s	f	s	f	s	f	s	f	s	f	s	f	s	f	s	f	s
			†		†		†	†	†	†	†	†		†	†	†	

56 De 16 forskellige udsagn

T 16 er speciel ved, at udsagnet er sandt uafhængigt af, om A og B er sande eller falske. T 16 kaldes en tautologi og kan f.eks. udtrykkes ved $A \vee \overline{A}$.
T 1 er derimod altid falsk og kaldes en absurditet. T 1 $= A \wedge \overline{A}$.

Opgave 14. Hvilke udsagn har de sandhedstabeller, der er markeret med pile på figur 56? De har været nævnt tidligere i teksten.

Opgave 15. Hvordan kan udsagnene mærket med pile udtrykkes ved hjælp af de tre tegn $\wedge$, $\vee$ og negation?

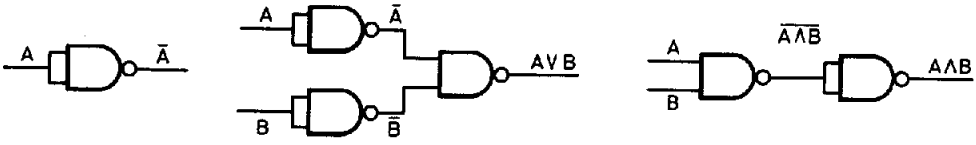
Opgave 16. Vis, at alle 16 udsagn på figur 56 kan udtrykkes ved hjælp af de tre tegn $\wedge$, $\vee$ og negation.

Opgave 17. Vis ved hjælp af sandhedstabeller, at $\overline{A} = \overline{A \wedge A}$.

Opgave 18. Vis på tilsvarende måde, at $A \vee B = \overline{\overline{A} \wedge \overline{B}}$.

Opgave 19. Vis, at $A \wedge B = \overline{\overline{A \wedge B}}$.

Ikke-og-porten kan således erstatte ikke-porten eller eller-porten og og-porten. Ikke-og-så ? Se figur 57 til 59.



57 Inverter af NAND. 58 OR GATE af NAND. 59 AND GATE af NAND.

Sammenholder man dette med resultaterne fra opgave 14 ses, at alle 16 udsagn kan udtrykkes ved ikke-og-porten, dvs. opbygges af NAND GATES. Man har i en kort periode opbygget store dele af datamater af ene NAND GATES for at drage fordel af masseproduktion af en enkelt komponent. Men nu har dette kun teoretisk interesse, fordi man integrerer mange logiske funktioner i en enkelt siliciumkrystal (chip).

Logisk algebra.

Der gælder en række regler for sammensætning af udsagn, som blev opstillet af George Boole. Man kalder dem regneregler for udsagn eller logisk algebra.

1	$A \vee \overline{A} = s$	7	$A \wedge s = A$	13	$A \wedge (A \vee B) = A$
2	$A \vee A = A$	8	$A \wedge f = f$	14	$A \vee (\overline{A} \wedge B) = A \vee B$
3	$A \wedge A = A$	9	$A \wedge B = B \wedge A$	15	$A \wedge (\overline{A} \vee B) = A \wedge B$
4	$A \wedge \overline{A} = f$	10	$A \vee B = B \vee A$	16	$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$
5	$A \vee f = A$	11	$B \wedge (A \vee \overline{A}) = B$	17	$\overline{A \vee B} = \overline{A} \wedge \overline{B}$
6	$A \vee s = s$	12	$A \vee (A \wedge B) = A$	18	$\overline{A \wedge B} = \overline{A} \vee \overline{B}$

Bemærk, at s står for en tautologi, og at f står for en absurditet.

Disse regler kan let eftervises ved hjælp af sandhedstabeller. Som eks. vil vi vise regel 17.

A	B	$A \vee B$	$\overline{A \vee B}$	$\overline{A}$	$\overline{B}$	$\overline{A} \wedge \overline{B}$
f	f	f	s	s	s	s
f	s	s	f	s	f	f
s	f	s	f	f	s	f
s	s	s	f	f	f	f

Af tabellen på figur 60 ses, at $\overline{A \vee B} = \overline{A} \wedge \overline{B}$.

60 Regel 17.

Regel 17 og 18 er særdeles anvendelige, når man vil omforme og forenkle logiske udtryk. De kaldes De Morgans regler efter den engelske matematiker og filosof Augustus De Morgan. Eftervis nogle af de øvrige regler.

Mængdediagrammer.

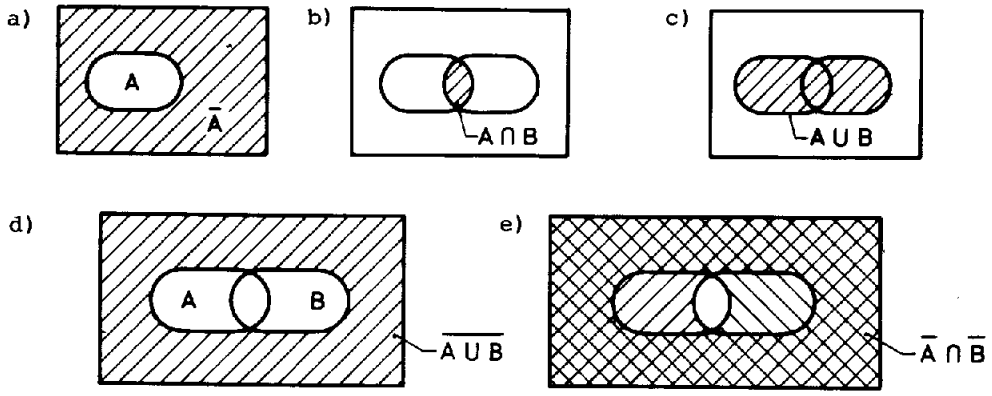
De nævnte regler for udsagn kan sammenlignes med de tilsvarende regler i mængdelæren. Bogstaverne er nu symboler for mængder. s erstattes af grundmængden G, f erstattes af den tomme mængde $\emptyset$, $\vee$ erstattes af $\cup$ (foreningsmængdetegn), $\wedge$ af $\cap$ (fællesmængdetegn) og negationen af komplementærmængdetegn, som vi her angiver på samme måde med en streg over symbolet.

De første fire logiske regler ovenfor bliver til:

1) $A \cup \overline{A} = G$ 2) $A \cup A = A$ 3) $A \cap A = A$ 4) $A \cap \overline{A} = \emptyset$

Opgave 20. Formuler de øvrige regler og vis, at de gælder for mængder.

Beviserne føres lettest ved at anvende de såkaldte Venn diagrammer, også kaldet bollefigurer. Figur 61 viser eksempler på Venn diagrammer.



61 Venn diagrammer til at vise, at $\overline{A \cup B} = \bar{A} \cap \bar{B}$.

Det ses, at det skraverede areal på figur 61d = det dobbeltskraverede areal på figur 61e, så regel 17: $\overline{A \cup B} = \bar{A} \cap \bar{B}$ er bevist. Mængdelæren og udsagnslogikken er således ækvivalente, dvs. beviser man en regel eller en sætning i mængdelæren, kan resultatet direkte overføres til udsagn og omvendt. Det er da snildt.

Følgende sammensatte udsagn: A v (C ∧ B) v C v (C ∧ A) kan omskrives til følgende mængdeudtryk: A U (C ∩ B) U C U (C ∩ A), som er lettere at overskue.

Opgave 21. Vis ved hjælp af mængdeboller, at mængden ovenfor kan reduceres til A U C. Udsagnet reduceres derfor til A v C.

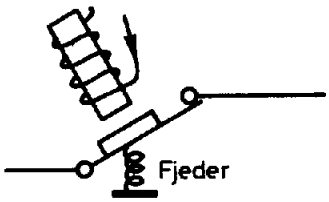
Opgave 22. Reducer $(\bar{A} \wedge B) \vee (A \wedge B)$ ved hjælp af Venn diagrammer. Prøv også at løse problemet ved hjælp af de logiske regler.

Den symbolske logik er et værktøj, hvorved man ved en forholdsvis ukompliceret teknik kan afgøre, om bestemte udtryk eller udsagn er identiske med nogle andre udsagn. Man kan f.eks. også se, om det ene udsagn følger af det andet. Mange betydelige filosoffer, f.eks. Bertrand Russel og Ludvig Wittgenstein, har i dette århundrede anvendt den til at analysere filosofiske problemer. Mange klassiske problemer (paradokser) er reduceret til problemer med at anvende (daglig)sproget rigtigt, således at de ikke længere regnes for at være filosofiske problemer, men sprogproblemer.

Det skal dog nævnes, at den symbolske logik har sine begrænsninger. Jo mere præcist man anvender sine symboler, jo mere begrænset bliver også deres anvendelsesområde. De kan ikke altid anvendes på menneskelige problemer, der gerne er mere nuancerede. Der findes også mellemværdier mellem falsk og sandt, ja og nej, eksisterer og eksisterer ikke.

De første regnemaskiner.

Ret hurtigt efter opdagelsen af ækvivalensen mellem symbolsk logik og elektriske netværk opstod den tanke: Man kan anvende logik til at løse netværksproblemer - kan man ikke også gøre det omvendte, altså anvende netværk til at løse logiske problemer, tælle, regne, etc.?
Umiddelbart før anden verdenskrig arbejdede den tyske ingeniør Conrad Zuse på at konstruere mekaniske maskiner, der kunne klare alle regningsarter. Det var de første almene maskiner, der regnede binært (i to-talsystemet). Den første maskine, Z1, var helt mekanisk. Senere benyttede Zuse relæer til konstruktion af regneenheden. I 1939 var denne første elektroniske regneenhed, bestående af 200 relæer, færdig. Ved at benytte det mekaniske lager fra Z1 havde han nu Z2-maskinen kørende. Den første egentlige relæmaskine, Z3, var klar i 1941, og det var verdens første programmerbare regnemaskine. Den indeholdt 2600 relæer. Zuse var også inde på den tanke at bruge elektronrør. Ideen blev forladt, da disse var for upålidelige og for dyre.

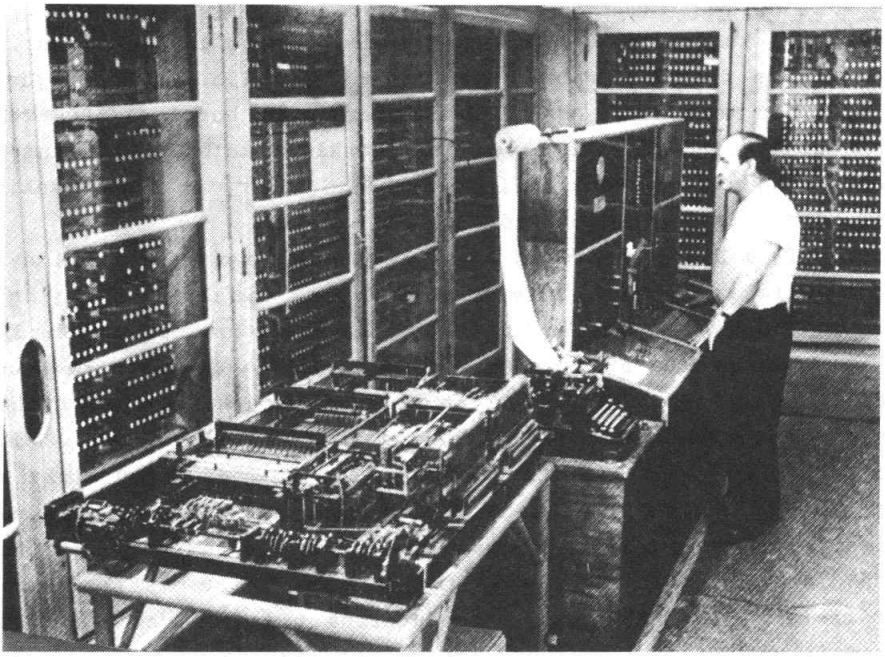


62 Relæ.

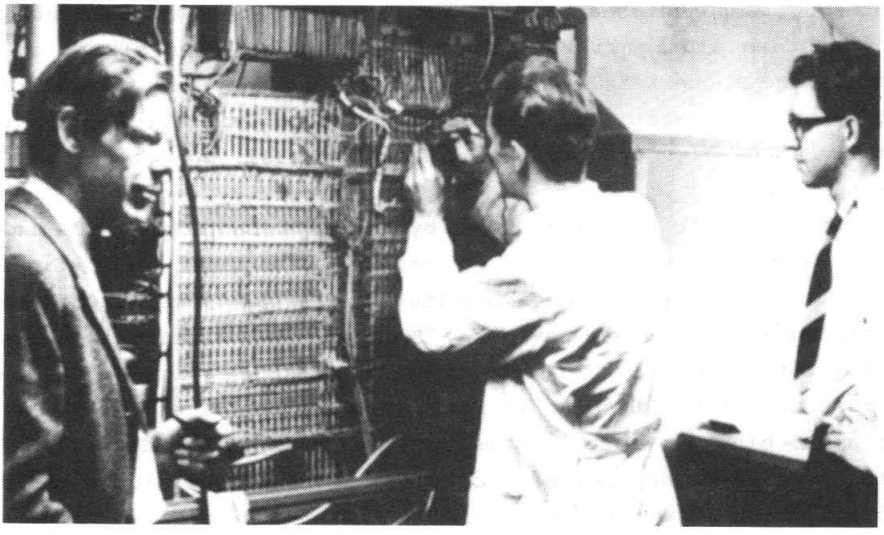
Relæer er langsomme. De kan højst skifte stilling 10 gange pr. sek. Maskiner med radiorør er 1000 gange hurtigere.
Figur 62 viser princippet i et elektromagnetisk relæ. En strøm i spolen på elektromagneten får kontakten til at slutte. Når der ingen strøm går i spolen, afbrydes kontakten ved hjælp af en fjeder.

Tyskernes arbejde fik imidlertid ikke nogen afgørende indflydelse på udviklingen af elektroniske regnemaskiner. Krigen kom, og de internationale relationer hørte op. Konstruktørene kunne ikke få statsstøtte i Tyskland. Hitlers regime var så sikker på at sejre hurtigt, at man ikke ville investere i noget, man ikke mente kunne bruges til noget, og som kunne tage år at udvikle.
I England var man hårdt presset ved krigens begyndelse. Militæret var underlegent, og man så sin chance i at foregribe modstanderens bevægelser ved hjælp af et effektivt efterretningsvæsen. Store summer og megen intelligens blev sat ind på at lave maskiner, der kunne lave

koder, der kunne ændres fra dag til dag, og at lave maskiner, der kunne bryde fjendens koder. Resultatet blev bl.a. en kodebrydningsmaskine med mere end 2000 radiorør, der meget hurtigt (5000 tegn/s) kunne analysere koder og bryde dem. Maskinen kaldtes Colossus efter dens omfang og var køreklar i 1943. Mange mener, at denne maskine var helt afgørende for krigens udfald.
Under krigen byggede Harvard University i USA sammen med IBM en mægtig regnemaskine af relæer, Harvard Mark I. Den var 2 1/2 m høj og 18 m lang og indeholdt mere end en million komponenter. Den var færdig i 44 og blev den sidste store relæmaskine. I USA havde man også brug for stor regnekapacitet bl.a. til at beregne projektilbaner. Efter 30 måneders arbejde i døgndrift havde man i 1946 computeren ENIAC i drift. Det var en almen regnemaskine, som rummede 19000 elektronrør, der krævede strøm fra et mindre elværk og havde et enormt stort køleanlæg. Maskinen kunne programmeres, men kun ved at flytte rundt på ledninger til maskinens forskellige dele ved stort omstillingsbord.
Det næste udviklingsspring i computerens historie var den geniale matematiker John von Neumanns ide om det lagrede program til at styre computeren. Programmeringen foregik, ikke ved at flytte ledninger, men ved at lave et program i to-talsystemet, og lagre det i maskinen sammen med de data, maskinen skulle behandle.
Der blev konstrueret flere computere, men omkostningerne var stadigvæk enorme og komponenterne upålidelige, og det var en udbredt antagelse, at computere aldrig ville komme til at spille nogen større rolle i almindelige menneskers tilværelse. Disse maskiner var forbeholdt stater og enkelte meget store firmaer og ville aldrig blive lavet i stort antal. Men så blev transistoren opfundet i 1947, og det ændrede billedet fuldstændigt.
Den første danske computer, blev bygget på foranledning af Akademiet for de Tekniske Videnskaber. Man oprettede herunder et institut ved navn "Regnecentralen, Dansk institut for Matematikmaskiner". Regnecentralen kopierede den svenske "BESK" (Binär Elektronisk Sekvens Kalkulator) opbygget af radiorør. Resultatet blev den første danske edbmaskine, DASK (Dansk Binær Sekvens Kalkulator), som var klar i 1957.
Det næste blev den transistoriserede computer, GIER (Geodætisk Instituts Elektron-Regnemaskine) produceret af Regnecentralen i samarbejde med Geodætisk Institut. Strukturen i GIER er designet af Torben Krarup og Bjarner Svejgaard (dengang ansat på Geodætisk Institut) sammen med det tekniske team på Regnecentralen. Prototypen stod klar i december 1960.



63 Zuses relæmaskine Z4, der var færdig i 1945 - den eneste, som overlevede bombningen af Berlin. Som Z1 og Z2 var maskinen udstyret med et mekanisk lager som ses i forgrunden. Maskinen fungerede efter en ombygning ved Zürichs Tekniske højskole i perioden 1955 til 60.



64 Fra venstre ses Bjarner Svejgård, Henning Isakson og Bent Scharøe Petersen foran GIER, som blev produceret af Regnecentralen i samarbejde med Geodætisk Institut. GIER var klar i 1961 og siden seriefremstillet.

2. Integrerede kredse

Oversigt

Hovedteksten indledes med en oversigt over transistoren og de integrerede kredsløbs historie. Dernæst omtales de to mest anvendte integrerede logiske familier: TTL- og CMOS-kredse.
Ved øvelserne testes virkemåden af TTL-kredsene NAND, NOR og NOT enkeltvis og sammensat til andre logiske funktioner som f.eks. en hukommelsescelle og en clock-generator.
I det supplerende stof gives en kort beskrivelse af fremstillingsteknik for de integrerede kredse, og det vises, hvordan en TTL-kreds er opbygget. Til sidst omtales tri-state logiske kredse.

Hovedtekst

De første transistorer blev fremstillet på Bell Telephone laboratorierne i USA i 1948. Det skete efter en intens målforskning, der var begyndt umiddelbart før anden verdenskrig. Man havde allerede da ideen om, at halvledere kunne erstatte radiorør i en forstærker. Arbejdet med at fremstille en halvlederforstærker blev indstillet under krigen og først genoptaget i 1945.
På jagten efter egnet halvledermateriale var man efterhånden nået frem til at anvende germanium og silicium. Man placerede metallet tæt op ad halvledermaterialet og varmede op, til metallet blev smeltet ind i halvlederen. Herved fik man en type transistorer, punkttransistorer, der dog ikke anvendes mere. (På engelsk hedder de junction-transistors.)
Omkring 1950 lærte man sig at inddampe fremmedatomer i germanium- eller siliciumkrystaller ved en planarteknik, der beskrives i det supplerende stof. Transistorerne, der anvendes i kapitel 1, er planartransistorer.
Transistoren blev få år efter dens opdagelse fremstillet i store mængder, og den blev meget billig. Da den er meget driftsikker og kun bruger lidt energi, skabte den revolution inden for elektronikken, som indtil da var baseret på radiorør. Disse var store og energikrævende. Shockley, en af hovedpersonerne bag fremstillingen af transistorer, sammenlignede engang radiorørs effektivitet med brugen af en lastbil til transport af et pund smør.

Transistorer er også langt mindre end radiorør, og da man samtidig formindskede dimensionerne af de øvrige elektroniske komponenter (f.eks. modstande og kondensatorer), blev resultatet som bekendt, at man kunne fremstille små transportable radiomodtagere, båndoptagere og fjernsynsapparater.
I hvert af disse apparater anvendes ret få transistorer (5-50 stk.), så på trods af, at der fremstilles millioner af radioer, er edb-branchen blevet langt den største forbruger af transistorer. Et digitalt regneværk skal ikke være ret stort, før der anvendes tusindvis af transistorer, og selv en mellemstor hukommelse kræver langt over hundredtusinde transistorer. Og transistorens tekniske udvikling og dens masseproduktion og billiggørelse er i høj grad bestemt af udviklingen inden for edb-teknikken.
De første datamaskiner med radiorør blev udviklet lige efter anden verdenskrig (se evt. kap.1, s.36).
Man havde store problemer med disse maskiners manglende driftsikkerhed. Selvom radiorør har en middellevetid på ca. 2500 timer, er der alligevel stor variation i de enkelte rørs levetid. Der opstår let fejl ved produktionen af nogle af rørene, uden at det kan konstateres, før de går i stykker. Hvis ikke alle døgnets 24 timer skulle anvendes til at udskifte rør, var de 19000 rør i en af de største maskiner nogenlunde det maximale, man kunne anvende, idet der i gennemsnit ville gå et rør i stykker hvert 8. minut.
Efter transistorens fremkomst blev datamaskinerne transistoriserede med enkelttransistorer og resistorer (modstande), der var forbundet med ledninger. Disse maskiner var langt mere pålidelige, da transistorer har en levetid, der er flere tusinde gange længere end levetiden for radiorør. Men der var stadig problemer med de hundredtusinde ledninger, der forbandt de enkelte komponenter. Der kunne opstå fejl ved monteringen, testarbejdet var omstændeligt, lodninger kunne være upålidelige etc.
Den endelige løsning på alle disse problemer var den integrerede teknik, hvor alt, transistorer, modstande, ledninger etc., blev indbygget i en enkelt lille siliciumkrystal, en såkaldt chip.
Allerede få år efter transistorens opfindelse begyndte en udvikling, hvor man lærte sig at bearbejde en siliciumkrystal ved en række processer, hvor man inddamper forskellige andre atomer efter tur. Ved hjælp af en fotografisk teknik fremstiller man masker, der kan placeres foran chippen, mens man inddamper urenhedsatomerne for at lave p- og n-halvledere, resistorer, ledere og isolatorer.

Den grundlæggende integrerede teknik var færdigudviklet omkring 1960, men den forfines stadig, således at mængden af de komponenter, der kan indlægges i hver chip, siden da er fordoblet hvert år. Der må selvfølgelig være en grænse for denne udvikling bl.a. bestemt af atomernes størrelse, men grænsen er langt fra nået endnu.
Udviklingen blev i høj grad finansieret af de militære bevillinger i USA. Man havde behov for pålidelige datamaskiner, der kunne bearbejde meldinger fra hundredvis af observationsposter over hele kloden, og maskiner, der kunne udforme forsvars- og angrebsstrategier etc. Da U.S.A. i første omgang havde svagere løfteraketter end Sovjetunionen, blev det også vigtigt at udvikle pålideligt, lidet energikrævende og småt digitalt udstyr til missiler og satellitter.
I begyndelsen af datamatens udviklingshistorie var der ingen kommerciel interesse for datamater. Selv ikke de mest fremsynede forretningsmænd var klar over, at computere var på nippet til at blive en kæmpebranche. Faktisk mente man, at verdensmarkedet for computere ville blive ganske lille, hvilket bl.a. ses af, at kun et kontorfirma gik ind i det, nemlig IBM.
Men med den integrerede teknik ændredes billedet fuldstændigt i løbet af ganske få år. De masseproducerede billige chips fandt en række nye anvendelser og blev grundlaget for en af den kapitalistiske verdens største vækstindustrier.
Datamaskinernes funktion ændredes efterhånden fra at være regnemaskiner (talknusere) til at blive informationsbearbejdere, databanker, overvågnings-, kontrol- og styringsmaskiner og legetøj.

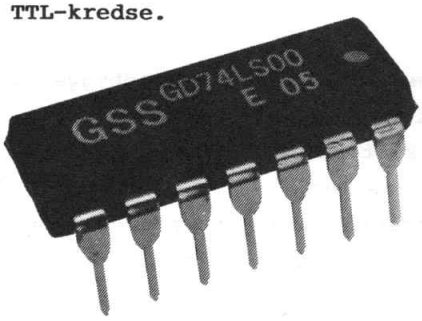
Integrerede kredse.

I tidens løb har man udviklet mange typer logiske kredse indbygget i en enkelt siliciumkrystal - en chip. Disse kredsløb kan inddeles i familier med ensartet opbygning, således at de inden for en familie kan sammenkobles efter ganske få, simple regler.
Man har tidligere anvendt integrerede kredse opbygget af dioder og transistorer, som de er beskrevet i kapitel 1. Dog har man videreudviklet dem således, at de foruden at virke logisk korrekt også opfylder visse sekundære krav, som f.eks. at de skal kunne reagere meget hurtigt og bruge meget lidt strøm.
Vi vil i dette kapitel omtale to kredsløbsfamilier, som i øjeblikket dominerer markedet, nemlig TTL-kredse og CMOS-kredse, der hver har deres fortrin.

TTL står for **T**ransistor **T**ransistor **L**ogic. Disse kredse er opbygget af transistorer af samme type, som vi anvendte i kapitel 1. De skifter hurtigt logisk tilstand (på mindre end 10 ns = 10^{-8} s). De er billige og robuste, så de bliver anvendt ved langt de fleste eksperimentelle øvelser i denne bog.

CMOS står for Complementarity Metal Oxide Semiconductor, hvor Complementarity henviser til, at der anvendes transistorer af forskellige typer i samme krystal. Derved kan man benytte transistorer, hvor man ellers ville benytte modstande, hvilket sparer plads. Metal Oxide henviser til, at MOS-transistorer er fremstillet ved brug af SiO_2. MOS-transistor er mindre strømkrævende, fylder mindre på en chip og er lettere og billigere at fremstille end TTL-kredse, så MOS-transistorer har specielt fundet anvendelse i store og komplicerede kredse som f.eks. mikroprocessorer og hukommelser. Til gengæld er CMOS-kredse mere følsomme over for f.eks. statisk elektricitet og forkert poling, og CMOS-kredsene er typisk 20 gange så langsomme til at reagere som TTL-kredse. Til gengæld er strømforbruget typisk 200 gange mindre.
Hver logisk familie er dog inddelt i underfamilier efter reaktionstid og strømforbrug, og generelt kan man sige, at hurtige kredse er meget strømforbrugende. Efterhånden er der dog kommet meget hurtige (og meget strømforbrugende) CMOS-kredse, ligesom der er udviklet meget lidt strømforbrugende (og langsomme) TTL-kredse.

TTL-kredse.



TTL-kredse fremstilles i (amerikansk) standard plastickapsler med 14, 16 eller flere ben i hver. Selve silicium-krystallen, som kredsene er opbygget på, fylder kun et par mm^2.

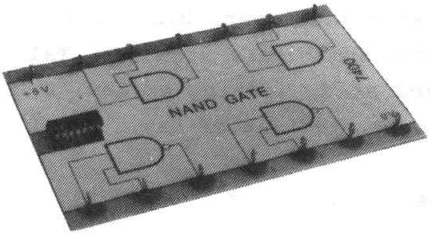
65 Billede af TTL-kredsen 7400.

Som eksempel er valgt TTL-kredsen 7400, der indeholder 4 stk. NAND GATES hver med 2 indgange og en udgang. De 4 logiske kredse har fælles spændingsforsyning 0 V og +5 V, men ellers virker de uafhængigt af hinanden.

Øvelser

TTL-kredse er lette at arbejde med, billige og temmelig robuste, så får du en ide til et kredsløb, kan du trygt gå i gang. Blot skal man være omhyggelig med at vende spændingsforsyningen rigtigt, ligesom man ikke må tvinge en udgang Høj/Lav med en ledning eller koble to udgange sammen.

I det følgende tænker vi os, at man benytter prøveplader til enten 14 eller 16 ben som vist på figur 66. Prøvepladen er blot en printplade med en IC-sokkel, hvor hvert ben er forbundet til et printspyd, så man let kan komme til at forbinde ledninger til de rigtige ben på IC'en. Desuden kan man anbringe en seddel over prøvepladen med benfordelingen til den IC, som er placeret i soklen.



66 Prøveplade til IC'ere. Sedlerne over benfordelingen findes side 111.

Man skal være meget omhyggelig med ikke at beskadige benene på IC'en, når den anbringes i soklen på prøvepladen, men specielt, når IC'en skal afmonteres, er det let at komme til at bøje eller knække benene. Så derfor kan det anbefales at lirke IC'en op af soklen med en skruetrækker.

I stedet for at arbejde med 5 V fra en spændingsforsyning, kan man ved øvelserne med en enkelt IC blot bruge et 4,5 V batteri.

Man tester, om udgangsspændingen er Høj eller Lav (sand eller falsk), med den samme slags udlæsemodul med lysdiode, som blev anvendt ved øvelserne i kapitel 1.

Øvelse 1. Test af NAND, NOR og NOT GATES.

Undersøg sandhedsværdierne i en (eller flere) NAND GATES i 7400 ved at forbinde indgangene til 0 eller 5 V. Lav en sandhedstabel. Nedskriv den evt. efter hukommelsen inden forsøget og se efter, om kredsen fungerer, som den skal. Virker en ikke forbundet indgang, som var den Høj?

Prøv evt. tilsvarende forsøg med 7402, der indeholder 4 stk. NOR GATES, og 7404, der indeholder 6 stk NOT GATES.

Øvelse 2. NAND som NOT GATE.

En NAND GATE kan forbindes således, at den virker som en NOT GATE. Prøv.

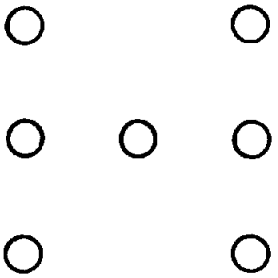
Øvelse 3. NAND som AND GATE.

Lav en AND GATE af 2 NAND GATES. Se evt. supplerende stof s. 33n.

Øvelse 4. NAND som OR GATE.

Lav en OR GATE af 3 NAND GATES. Se evt. supplerende stof s. 33n.

Øvelse 5. Terning.



67 "Terning".

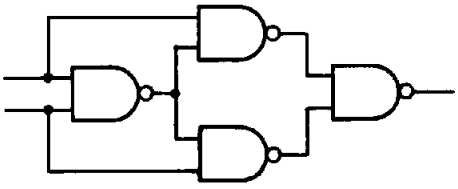
Med et logisk kredsløb og 7 lysdioder skal du styre visningen af en terning. Lysdioderne placeres som vist på figur 67. Der skal være 7 ledninger fra det logiske kredsløb - en til hver sin lysdiode på terningen. Indgangene består af 3 ledninger, hvor tallene fra 1 til 6 sendes som binære signaler. Tallet 2 angives ved at 1. ledning er Lav, 2. ledning er Høj og 3. ledning er Lav.

1 = 001 = LLH
2 = 010 = LHL
3 = 011 = LHH
4 = 100 = HLL
5 = 101 = HLH
6 = 110 = HHL

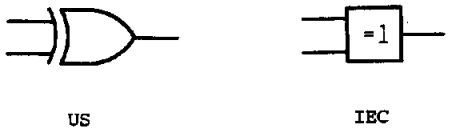
Hvilket indgangssignal svarer til udgangssignalet til lysdioden i midten?
Hvor mange forskellige udgangssignaler skal der være?
Lav det nødvendige kredsløb og afprøv det.
Husk at lysdioder skal have en beskyttelsesmodstand på 220 Ω.
Kan terningen også vise tallene 0 og 7?

Øvelse 6. EXCLUSIVE-OR.

Opbyg en EXCLUSIVE-OR GATE = EX-OR, hvor udgangen kun er Høj, hvis de to indgange er forskellige. EX-OR kan laves af 4 NAND GATES. Se figur 68. Test, om kredsløbet virker, som det skal.



68 EXCLUSIVE-OR af 4 NAND GATES.



US IEC

69 Symboler for EXCLUSIVE-OR.

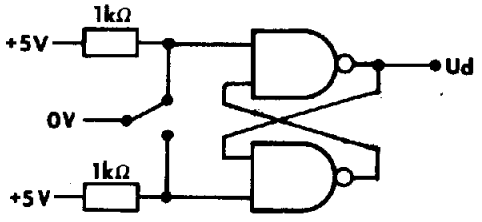
Lav en sandhedstabel for en EX-OR GATE.

Øvelse 7. De Morgans regler.

Test De Morgans sætninger: $\overline{A \wedge B}$ er ækvivalent med $\overline{A} \vee \overline{B}$, og $\overline{A \vee B}$ er ækvivalent med $\overline{A} \wedge \overline{B}$. (To udsagn er ækvivalente, hvis de har samme sandhedstabel.)

Øvelse 8. Prelfang - en støjfri kontakt.

Ved test af simple portkredse kan man tildele indgangene sandhedsværdier ved simpelthen at forbinde dem til 0 eller 5 V, evt. med en mekanisk kontakt. En sådan kontakt slutter og afbryder ikke øjeblikkelig. Den kan give en række støjspændinger, som mere komplicerede TTL-kredse, f.eks. tællere og registre, kan nå et reagere på. En mekanisk kontakt kan således ikke anvendes her.
I stedet kan man anvende et prelfang - en støjfri kontakt, som vist på figur 70.



70 Prelfang.

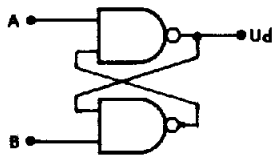
Den mekaniske kontakt er her forbundet til to NAND GATES.
En indgang på hver af disse NAND GATES er forbundet til 5 V via en modstand. Med mindre disse indgange tvinges til at blive Lav med kontakten, er de Høje.
Tvinges indgangen på NAND GATE 1 til at blive Lav med kontakten, vil udgangen på NAND GATE 1 blive Høj (ligegyldig hvad den anden indgang er).

Udgangsspændingen fra NAND GATE 1 føres til indgangen på NAND GATE 2, hvor begge indgange nu er Høje, og dens udgang vil blive Lav. Denne udgangsspænding føres tilbage til indgangen på NAND GATE 1, men det ændrer intet. Situationen er stabil.
Afbrydes forbindelsen mellem 0 V og indgangen på NAND GATE 1, får det ikke NAND GATE 1 til at ændre udgangsspænding. Den forbliver Høj, fordi den anden indgang er Lav. Det betyder således ikke noget, hvis den mekaniske kontakt ikke er helt perfekt. Først når kontakten skifter, så indgangen på NAND GATE 2 bliver Lav, ændres udgangen på NAND GATE 1 til Lav.

Opbyg en støjfri kontakt og test den.

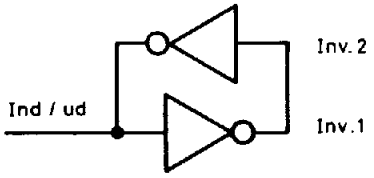
Øvelse 9. Flip-flop - en hukommelsescelle.

Da en ikke-forbundet indgang på en TTL-kreds er Høj, vil kredsen, der er vist på figur 71, virke ligesom den støjfri kontakt. Den vil huske, hvilken indgang der sidst har været Lav. Den virker altså som en hukommelse. Kredsen kaldes en flip-flop. Test den.



71 Hukommelsesenhed af 2 NAND GATES.

Man har lavet mange forskellige typer af flip-flops, der anvendes i registre, tællere og hukommelser. På figur 72 vises en særlig simpel flip-flop, som vi anvender i vores model af en hukommelse i kapitel 4.



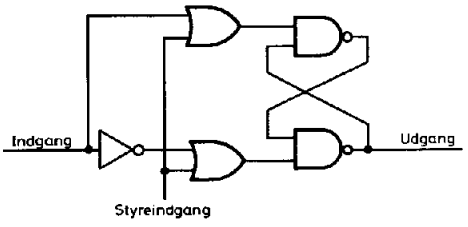
72 Hukommelsesenhed af 2 Invertere.

Den består af to invertere, der har fælles ind- og udgang. Tvinges indgangen Høj, bliver udgangen af inverter 1 Lav og udgangen af inverter 2 Høj. Tvinges indgangen Lav, bliver udgangen af inverter 1 Høj og udgangen af inverter 2 Lav.

I begge tilfælde bliver udgangsspændingen som indgangsspændingen, og begge tilstande er stabile.

Denne flip-flop vil huske, hvad indgangen sidst har været. Test den med kredsen 7404.

Øvelse 10. Latch.



En latch er blot en hukommelsesenhed med en styreindgang, med hvilken man vælger, hvornår signalet på indgangen skal huskes. En latch kan være opbygget som vist på figur 73. For at forstå virkemåden, kan du gennemgå følgende argumenter:

73 Latch, hukommelse med styreindgang.

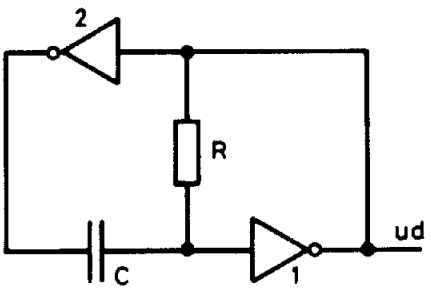
Indse ved at afprøve de forskellige muligheder, at når indgangene til flip-floppen er forskellige, sættes dens udgang til den øverste indgangsværdi.

Indse, at indholdet af flip-floppen er stabilt, når begge de to indgange er Høje.
Indse, at OR GATENES udgange begge er Høje, når styreindgangen er Høj. Er styreindgangen Lav, er udgangene af OR GATENE forskellige, sådan, at udgangen foroven har samme værdi som indgangen til latch'en.

Derfor sættes latch'en, når styreindgangen er Lav, og den husker signalet, når styreindgangen er Høj.

Øvelse 11. Astabil flip-flop. Clock-generator.

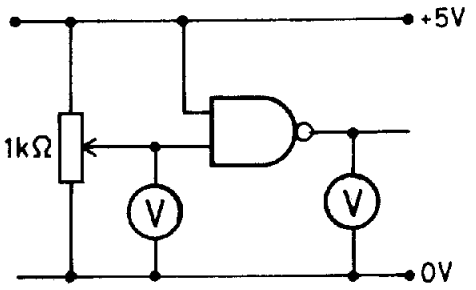
I en mikrodatamat styres tidssynkroniseringen af en clockimpulsgenerator, ofte blot kaldet en clock-generator. Normalt anvendes en siliciumkrystal, der f.eks. svinger med 4 MHz (4 mio. impulser pr. sekund).
Vi anvender en astabil flip-flop som vist på figur 74.
Først vises, at denne flip-flop, der også indeholder en modstand og en kondensator, er ustabil. Man siger, at den er astabil.



74 Clock-generator.

Tænker vi os f.eks., at indgangen på inverter 1 er Lav, vil dens udgang være Høj og udgangen på inverter 2 og dermed også venstre kondensatorplade være Lav (0 V). Men denne tilstand forstyrres af, at der løber strøm gennem modstanden, da spændingen forneden er 0 V og foroven 5 V. Herved bliver højre kondensatorplade opladet og indgangsspændingen på inverter 1 bliver Høj.
Alle andre spændinger skifter. Nu bliver spændingen 0 V foroven ved modstanden og 5 V forneden. Kondensatoren bliver afladet gennem modstanden, indtil indgangsspændingen på inverter 1 igen bliver Lav. Og sådan bliver det ved. Frekvensen bestemmes af størrelsen af kondensatoren og modstanden.
En modstand på R og en kapacitans på C giver en frekvens på ca. 0,6/(RC).

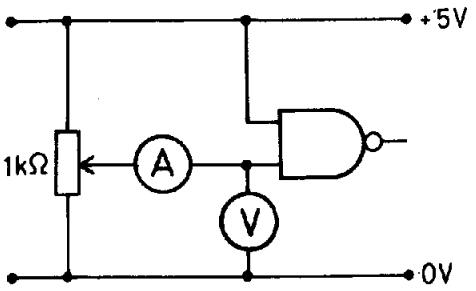
Øvelse 13. Spændingskarakteristik for en TTL-kreds.



75 Spændingskarakteristik af NAND GATE.

Vil man optage en udgangs/indgangs-spændingskarakteristik for en TTL-kreds, kan man anvende et kredsløb, som vist på figur 75. Her kan indgangsspændingen på en af indgangene, f.eks. på en NAND GATE, varieres fra 0 V til 5 V ved hjælp af en skydemodstand. Denne kan f.eks. være på 1 k Ω. Man måler sammenhørende værdier af U_{ind} og U_{ud} og laver en graf, der viser, hvordan U_{ud} afhænger af U_{ind}. Beskriv denne sammenhæng i ord. Undersøg, om kredsen holder specifikationerne, der stilles til TTL-kredse. Se supplerende stof side 52n.

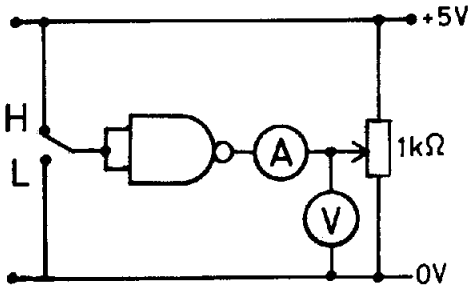
Øvelse 14. Indgangsstrøm ved TTL-kredse.



76 Indgangskarakteristik af en NAND GATE.

Figur 76 viser et kredsløb, hvormed man kan måle sammenhørende værdier af indgangsstrøm og indgangsspænding i en NAND GATE. Amperemetret skal kunne måle milli- og mikroampere. Som Voltmeter anvendes et digitalinstrument med høj indre modstand. Lav en serie målinger. Veksler strømretningen ved Høj og Lav indgangsspænding, som beskrevet i supplerende stof side 51?

Øvelse. 15. Udgangsstrøm ved TTL-kredse.



77 Udgangskarakteristik af en NAND GATE.

Figur 77 viser et kredsløb, hvormed man kan måle sammenhørende værdier af udgangsstrøm og udgangsspænding, når indgangene enten er Høje eller Lave. Her må udgangsstrømmen dog ikke overstige 120 mA. Er strømme og spændinger som beskrevet i supplerende stof side 51 og 52?

Supplerende stof

Den integrerede teknik, hvor man indbygger mange logiske kredsløb i en enkelt siliciumkrystal, har haft afgørende betydning for digitalelektronikkens udvikling. Med denne teknik kan man masseproducere pålideligt og billigt apparatur, der bl.a. kan bruges til registrering, styring og kommunikation.
Store menneskelige og økonomiske ressourcer er sat ind på at udvikle den integrerede teknik med kredsløb, der kan mere på mindre plads. I den sidste snes år er mængden af logiske funktioner placeret på en enkelt chip nogenlunde fordoblet hvert år. De største integrerede kredse indeholder nu mere end en million logiske funktioner. Selvfølgelig kan denne udvikling ikke fortsætte (atomerne har en endelig størrelse), men grænsen ser ikke ud til at være nået endnu, selvom de enkelte transistorer i chippen ikke længere kan ses i et lysmikroskop, når talen er om de højt integrerede kredse.
Det er bemærkelsesværdigt, at de enkelte kredsløb, der indlægges, stadig kan opfattes som opbygget af gammelkendte elektroniske komponenter, som transistorer, dioder og modstande. Men det er ikke sådan, at nye kredse først opbygges af enkeltkomponenter og afprøves. Det er langt billigere at beskrive de enkelte komponenter med en matematisk model, der indeholder de væsentligste karakteristiske egenskaber ved komponenterne, og så simulere de kredsløb, man ønske at bygge, på en computer. Denne beregner kredsløbets egenskaber, hvorpå man kan ændre kredsløbet og efterhånden finde frem til den bedste løsning.
Det skal dog bemærkes, at de enkelte komponenter, f.eks. transistoren, får andre egenskaber, når størrelsen bliver mindre end en mikrometer. De små afstande ændrer på tids- og strømforbruget. Den model, man anvender ved beregningerne, må naturligvis tilpasses herefter. Og selvom man gerne ville opbygge de højtintegrerede kredse af enkeltkomponenter, kunne man ikke.
En computer er altså nødvendig for at udvikle nye computere, og når man har fundet frem til et kredsløb, som man vil bygge, kan computeren også tegne positionen for de enkelte kredsløbselementer i en passende stor skala, med stor præcision.
Computeren anvendes også til at fastlægge positionerne af det tredimensionale mønster, der frembringer de elektroniske kredsløb.
Men inden vi beskriver dette nærmere, vil vi se på siliciumkrystallens egenskaber og forklare, hvordan de enkelte elektroniske komponenter kan fremstilles heraf.

Opbygning af elektroniske komponenter i silicium.

De integrerede kredse fremstilles ud fra helt rent silicium, der fremstilles ved en særlig smeltningsteknik. Af den rene silicium trækkes en perfekt enkelt-krystal i form af en stang, der typisk er 5 cm i diameter. Stængerne skæres i ca 1 mm tynde skiver, der renses og poleres, og da de fleste integrerede kredse kun fylder et par mm^2, fremstilles der flere hundrede kredsløb samtidigt af en skive silicium.

Som vi tidligere har beskrevet, er det først, når der i den rene silicium bliver indført en række fremmedatomer, at vi får halvledermaterialer, der kan anvendes til opbygning af dioder og transistorer.

Af silicium fremstilles en af de bedste isolatorer, man kender. Når krystallen opvarmes til ca. 1000 °C i en atmosfære bestående af ilt og vanddampe dannes et lag SiO_2 (kvarts). Tykkelsen bestemmes af temperatur og tid og kan kontrolleres nøje.

Man kan så fjerne laget igen, hvor man ønsker det, ved først at dække den oxiderede chip med en slags lak, der hærdes af ultraviolet lys. Lægger man en ugennemsigtig maske over den del, hvor kvartslaget ønskes fjernet, og belyser med UV-stråling, vil kun denne del blive ætset, når chippen bades i flussyre. Tilbage bliver den SiO_2, som ligger under det belyste lak. Lakken kan så fjernes igen.

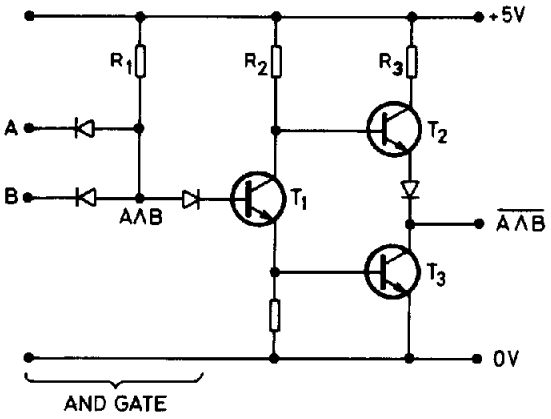
SiO_2 anvendes også, når krystallen visse steder skal tilføres fremmedatomer for at danne n- og p-materialer. Disse fremmedatomer kan tilføres på to måder. Ved diffusionsmetoden opvarmes krystallen med SiO_2 pålagt bestemte steder til ca. 1000 °C i en atmosfære af f.eks. boratomer eller phosphoratomer. Disse diffunderer da langsomt ind i den rene krystal, men næsten ikke i de områder, der er belagt med SiO_2. Ved ionimplantations-metoden omdannes de ønskede atomer først til ioner, hvorefter de med stor fart skydes ind i krystallen i et bestemt mønster ved hjælp af en ionaccelerator. Denne metode er meget præcis og anvendes mere, jo tættere de elektroniske komponenter kommer til at ligge.

Opbygning af TTL-kredse.

TTL-kredsene blev introduceret i 1964 af firmaet Texas Instruments og har siden været den mest dominerende logiske kredsløbsfamilie. TTL-kredsene er siden produceret af mange firmaer i lidt forskellige udgaver, men med de samme tekniske specifikationer. Man er således ikke afhængig af leverancerne fra et enkelt firma.

Som eksempel har vi valgt 7400, der indeholder 4 stk. NAND GATES.

De øvrige TTL-kredse fungerer logisk anderledes, men er opbygget efter de samme principper som 7400.



78 Opbygning af en NAND i 7400.

På figur 78 ses opbygningen af en enkelt NAND, og som det fremgår, indeholder kredsløbet 2 dioder og en modstand koblet som en AND GATE på samme måde som beskrevet i kapitel 1. Den ekstra diode mellem AND GATE'n og transistor T_1 er medtaget for at få basisspændingen på T_1 nær 0 V, ved Lav indgangsspænding. I praksis er de tre dioder opbygget som en enkelt transistor, hvor to emittere er koblet til den samme basis.

Transistoren T_1 definerer indgangsniveauet for Høj og Lav spænding, og transistorerne T_2 og T_3 er medtaget for at give veldefinerede udgangsspændinger og hurtige skift.

Først ser vi på den situation, hvor begge indgange er Lave. Da vil der gå strøm gennem dioderne i AND GATE'n - og dermed gennem R_1 - og udgangen af AND GATE'n bliver Lav. Det bevirker, at transistoren T_1 afbrydes, og dermed bliver basis på transistoren T_2 Høj, og T_2 kortslutter udgangen til +5 V gennem modstanden R_3. Der kan trækkes strøm ud gennem udgangen. Dog max 0,4 mA. Transistoren T_3 er samtidig afbrudt, så den ikke forstyrrer udgangsspændingen.

Hvis kun den ene indgang er Lav, vil den trække udgangen af AND GATE'n Lav, og resten vil være som før.

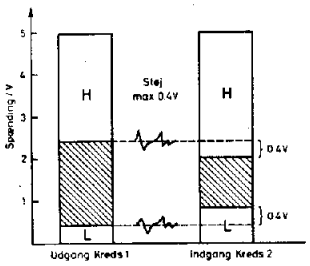
Når begge indgange er Høje, bliver udgangen af AND GATE'n Høj, og transistor T_1 bliver nu ledende. Dermed bliver basisspændingen > 0,7 V i forhold til emitterspændingen for T_3, og modsat for T_2, på grund af spændingsfaldet på 0,7 V over dioden mellem T_2 og T_3. T_3 kortslutter udgangen til 0 V, og T_2 afbrydes. Der kan nu trækkes en strøm ind gennem udgangen til 0 V, så udgangen bliver Lav. Dog max 16 mA for ikke at brænde T_3 af.

Alt i alt kan man sige, at transistoren T_1 benyttes til at adskille Høj og Lav, at T_2 benyttes til at trække udgangen af NAND GATE'n Høj, og at T_3 benyttes til at trække udgangen Lav.

Gøres en indgang Høj, går der max 0,04 mA ind gennem indgangen, og gøres indgangen Lav, trækkes der max 1,6 mA ud af indgangen. Sammenholder vi det med, at en udgang max kan levere en strømstyrke på 0,4 mA ved Høj spænding og max kan trække en strøm på 16 mA ved Lav spænding, ser vi, at man max kan koble 10 gates parallelt til en og samme udgang. Derimod kan man i princippet koble uendeligt mange gates i serie efter hinanden.
Det foregående omhandler standard-TTL-kredse, men der findes som nævnt flere TTL-familier. Low power Schottky TTL-kredse (LS-kredse) er en TTL-familie, der er lidt dyrere at fremstille, men som bruger mindre strøm og reagerer hurtigere end standard-TTL-kredse. En standard-kreds kan trække op til 40 LS-kredse parallelt, men til gengæld kan en LS-kreds kun trække 5 standard-kredse parallelt. En LS-kreds kan trække op til 20 LS-kredse. MOS-kredsene, som benyttes i store og komplicerede kredsløb, f.eks. i mikroprocessorer, bruger langt mindre strøm end TTL-kredsene, men for at kunne benytte MOS-kredse sammen med TTL-kredse, er de fleste MOS-kredse "TTL-kompatible", hvilket vil sige, at en MOS-kreds mindst kan trække en standard-TTL-kreds eller 4 LS-kredse på hver udgang.

Reaktionstiden for en portkreds er en vigtig faktor, når man vil lave logiske kredsløb, og for standard-TTL-kredse går der 10 ns = 10^{-8} sek, fra man skifter spænding på en indgang, til udgangen har skiftet spænding. Den tilsvarende tid er 200 ns for de fleste MOS-kredse. Den hurtigste TTL-type er Fast-TTL med en reaktionstid på 3 ns. Tilsvarende er den hurtigste MOS-type High-speed-CMOS med en reaktionstid på 10 ns.

Spændingsværdierne Høj og Lav i TTL-kredse.
Fabrikanterne angiver, at hvis forsyningsspændingen holdes mellem 4,75 V og 5,25 V, vil spændinger over 2 V på indgangene blive opfattet som Høj og spændinger under 0,8 V blive opfattet som Lav.



79 Spændingsniveauer i TTL-kredse.

Derimod vil en Høj udgangsspænding altid være større end 2,4 V og en Lav udgangsspænding altid under 0,4 V. Det vil sige, at man kan tillade et støjsignal udefra på op til 0,4 V, uden at et signal fra en gate mistolkes af den efterfølgende gate. Se figur 79.

Tri-state-logik.

Man må normalt ikke koble udgangene fra to TTL-kredse sammen, da der opstår konflikt, hvis den ene er Høj, og den anden er Lav.
Hvor flere signaler på skift skal transmitteres gennem den samme ledning, klares problemet ved at anvende såkaldte tri-state logiske kredse. Disse har en styreindgang, hvormed kredsens udgang kan afbrydes.
Som eksempel kan vi se på TTL-kredsen 74126, der indeholder 4 stk. tri-state buffere. Betegnelsen tri-state kommer af, at udgangen har tre tilstande: Høj, Lav og afbrudt.

80 Tri-state buffer.

På figur 80 er er vist symbolet for en af tri-state-bufferne fra IC'en 74126. Når styreindgangen er Lav, er forbindelsen til udgangen afbrudt, og når styreindgangen er Høj, virker kredsen som en almindelig buffer. En buffer er et kredsløb, hvor udgangsspændingen er lig med indgangsspændingen. En buffer anvendes derfor til at forstærke signaler med.

3. Regning med binære tal

Oversigt

I hovedteksten indføres de binære tal og sammenlignes med vore sædvanlige tal i ti-talsystemet. Desuden omtales addition og multiplikation af binære tal sammen med principperne for opbygningen af et kredsløb til addition af binære tal.
I øvelserne afprøves kredsløb til binær addition, adderen 7483 og ALU'en 74181.
Det supplerende stof indeholder foruden en omtale af binær subtraktion og division også en omtale af, hvordan man angiver ti-potenser, og hvordan man omsætter tal mellem forskellige talsystemer.

Hovedtekst

Vi skriver normalt vore tal i ti-talsystemet. Det er et positionssystem - det vil sige at tallenes position er afgørende for tallenes størrelse. F.eks. er tallene 1003 og 0013 forskellige. Det første er sammensat af 1 stk. tusinder og 3 stk. enere, og det næste tal er sammensat af 1 stk. tier og 3 stk. enere. Helt til højre angiver vi antallet af enere, dernæst antallet af tiere, hundreder, tusinder osv. Ved decimaltal sættes et komma til højre for enernes plads. Til højre for kommaet angives antallet af tiendedele, hundrededele, tusindedele osv. Alle de sædvanlige tal kan på den måde skrives ved hjælp af cifrene 0, 1, 2, 3, 4, 5, 6, 7, 8 og 9 - altså 10 forskellige cifre, og derfor kalder vi vores talsystem for ti-talsystemet.

Tilsvarende kan tallene skrives i to-talsystemet, hvor alle tal kan angives med cifrene 0 og 1. Tallet 19_{10} skrives som 10011_2 = 1·16 + 0·8 + 0·4 + 1·2 + 1·1. Generelt skriver man helt til højre antallet af enere (0 eller 1). Dernæst antallet af toere, antallet af firere, antallet af ottere osv. Tallet med cifrene $a_4\,a_3\,a_2\,a_1\,a_0$ betyder i to-talsystemet: $a_4 \cdot 2^4 + a_3 \cdot 2^3 + a_2 \cdot 2^2 + a_1 \cdot 2^1 + a_0 \cdot 2^0$.

Eksempel 1.
Tallet 49 kan opfattes som 32 + 16 + 1 eller $2^5 + 2^4 + 2^0$ og dermed skrives tallet 49_{10} som 110001_2.

$2^0 = 1$	$2^6 = 64$	$2^{12} = 4096$	$2^{18} = 262144$
$2^1 = 2$	$2^7 = 128$	$2^{13} = 8192$	$2^{19} = 524288$
$2^2 = 4$	$2^8 = 256$	$2^{14} = 16384$	$2^{20} = 1048576$
$2^3 = 8$	$2^9 = 512$	$2^{15} = 32768$	$2^{21} = 2097152$
$2^4 = 16$	$2^{10} = 1024$	$2^{16} = 65536$	$2^{22} = 4194304$
$2^5 = 32$	$2^{11} = 2048$	$2^{17} = 131072$	$2^{23} = 8388608$

Opgave 1. Skriv de første 32 tal i to-talsystemet.
Omskriv følgende tal fra ti-talsystemet til to-talsystemet:

a) 37 b) 100 c) 48

Omskriv også følgende tal fra to-talsystemet til 10-talsystemet:

a) 100000 b) 100 c) 1101001100

Addition af tal.

Udfør additionen:

$$\begin{array}{r} 107 + \\ \underline{279} \end{array}$$

Forhåbentlig fremkommer resultatet 386 ved først at addere enerne 7 og 9, der giver summen 16 (sum på 6 og 1 i mente). Dernæst adderes tierne inklusiv menten fra før etc.

I to-talsystemet foregår additionen analogt:

Eksempel 2.

$$\begin{array}{r} 111 + \\ \underline{1111} \\ 10110 \end{array}$$

Først adderes enerne: $1 + 1 = 2_{10} = 10_2$, altså en sum på 0 og en mente på 1. Dernæst adderes toerne, 1 + 1 og 1 i mente = $3_{10} = 11_2$ (sum på 1 og 1 i mente) etc.

Opgave 2. Oversæt tallene ovenfor til ti-talsystemet og kontroller resultatet.

Opgave 3. Udfør følgende additioner i to-talsystemet. Omskriv dernæst tallene til ti-talsystemet og kontroller resultaterne.

a) 11111 + 1 b) 1000011 + 1000011

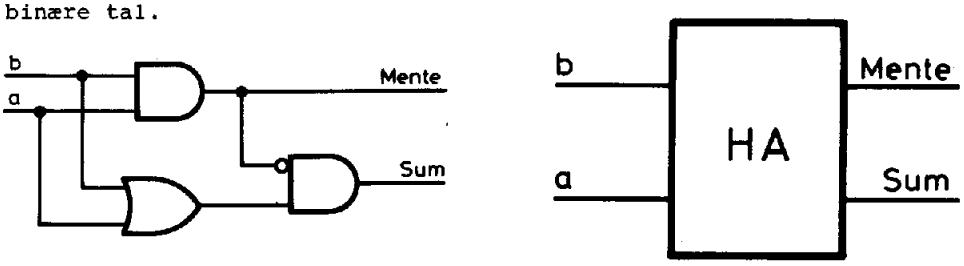
Additionsmaskine i to-talsystemet.

Som vi har set, adderes enerne efter følgende tabel:

				Mente	Sum
0	+	0	=	0	0
0	+	1	=	0	1
1	+	0	=	0	1
1	+	1	=	1	0

Hvis vi i stedet for tallene 1 og 0 tænker på Høj og Lav, har vi brug for to gates - en til at give summen og en til at give menten. Den gate, der giver summen, kaldes en EXCLUSIVE-OR og kan f.eks. laves af to AND, en NOT og en OR GATE. Hvad er det for en gate, der giver menten?

Opgave 3. Vis, at kredsløbet på figur 81 kan addere to encifrede binære tal.

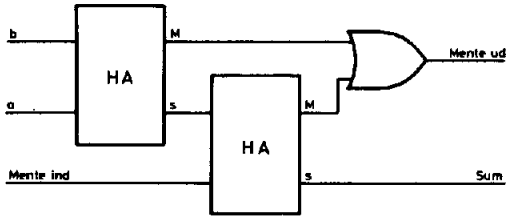


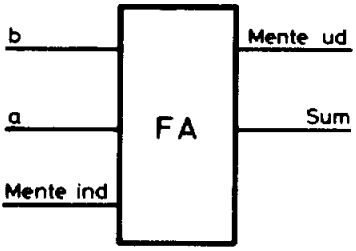
81 a) Half Adder af GATES. b) Symbol for Half Adderen.

En sådan sammensætning af gates til addition af to binære tal kaldes en Half Adder eller en HA og angives med symbolet vist på figur 81. Når to flercifrede tal skal adderes, får vi brug for også at kunne addere en mente. Et kredsløb, der kan addere to encifrede binære tal og en mente kaldes en Full Adder eller en FA. Den må fungere efter følgende tabel:

A	B	mente-ind	mente-ud	sum
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

Dette kan vi lave med følgende logik:

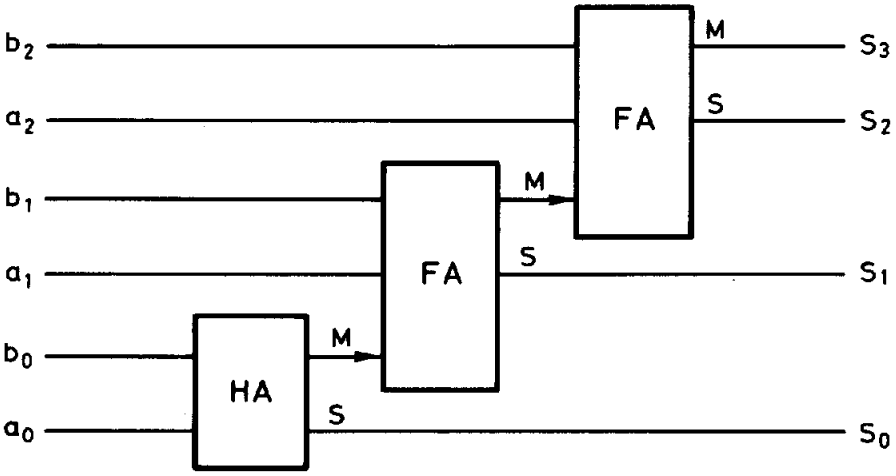




82 Full Adderen opbygget af Half Addere. b) Symbol for Full Adderen.

Opgave 4. Vis ud fra kendskabet til Half Adderen, at Full Adderen virker som ønsket.

Nu kan vi addere flercifrede tal. Ønskes to tre-cifrede binære tal adderet, kan det ske elektronisk med følgende kredsløb:



83 Kredsløb til addition af 2 tre-cifrede binære tal.

Opgave 5. Konstruer en additionsmaskine, der kan klare to otte-cifrede binære tal. Hvor mange indgange og udgange skal der være? Hvor mange gates medgår ved dit design?

Multiplikation af tal.

I vort sædvanlige talsystem er 3 · 4 blot en summation 4 + 4 + 4 = 12. Således kan vi naturligvis også gøre i to-talsystemet. F.eks. er 11 · 1001 = 1001 + 1001 + 1001 = 11011.

Opgave 6. Vis, at udregningen i to-talsystemet er rigtig.

Ved multiplikation af flercifrede tal udnytter vi normalt positionssystemet, som det fremgår af følgende eksempel:

$$\begin{aligned} 122 \cdot 244 = {} & 488 + (2 \cdot 244) \\ & 488 + (20 \cdot 244) \\ & 244 \quad\quad (100 \cdot 244) \\ & 29768 \end{aligned}$$

Altså ved omskrivningen 122 · 244 = (2 + 20 + 100) · 244, samt ved at udnytte, at man ganger med 10, 100, 1000 osv. ved blot at flytte tallet en, to, tre osv. pladser til venstre.

Det samme kan man gøre i to-talsystemet, og her er udregningerne endnu simplere, som det fremgår af følgende eksempel:

$$\begin{aligned} 10011 \cdot 1111 = {} & 1111 + (1 \cdot 15) \\ & 1111 + (2 \cdot 15) \\ & 0000 + (0 \cdot 15) \\ & 0000 + (0 \cdot 15) \\ & 1111 + (16 \cdot 15) \\ & 100011101 \end{aligned}$$

Opgave 7. Lav udregningen i ti-talsystemet og kontroller resultatet.

Opgave 8. Vis, at når man ganger et tal med 2 og 4 = 2^2 i to-talsystemet, svarer det blot til, at man rykker tallet 1 hhv. 2 pladser til venstre.

Det at gange i det binære talsystem indebærer et antal skift til venstre af tallene og et antal additioner, og derfor er der i regneenheden i computere indbygget specielle instruktioner til dette.

Opgave 9. Lav følgende udregninger i to-talsystemet:

a) 110 · 11 b) 1000 · 1000001 c) 1011 · 111

Øvelser

Øvelse 1. Adderen 7483.

4-bit Full Adderen 7483 monteres på en 16-bens prøveplade, og en seddel over adderens benfordeling lægges på pladen. Giv kredsen forsyningsspænding og forbind de 4 udgange Σ_1 - Σ_4 til et udlæsemodul. Før adderen kan benyttes, må den indkommende mente c_{ind} forbindes til 0 V med en ledning. (c står for carry = mente).

Nu kan man vælge de tal, man ønsker adderet, med indgangene A_1 til A_4 og B_1 til B_4.

Prøv først med 1 + 1, altså A_1 og B_1 til Høj og resten Lave.

Prøv med andre tal. Hvad sker der, når summen er større end 15? Forbind også udgangen c_{ud} til et udlæsemodul.

Evt. kan du prøve at koble to 7483-addere sammen ved at forbinde den udgående mente c_{ud} fra den første IC til den indgående mente c_{ind} på den næste. Derved kan man addere to otte-cifrede tal. Hvor mange udgange er der?

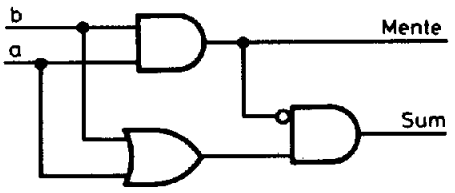
Øvelse 2. Udlæsning med et syvsegment-display.

Et syvsegment placeres på en 14-bens prøveplade, og en seddel med benfordelingen lægges på pladen. Syvsegmentet består, som navnet siger, af 7 segmenter, og med dette skrives alle tallene fra 0 til 9 ligesom på en lommeregners display. Hvert segment er forbundet til et ben og til den fælles anode (ben 14). Da segmenterne blot er lysdioder, som max kan tåle 2 V, skal der sættes en modstand i serie med segmenterne. Det gøres lettest ved at forbinde ben 14 i serie med en modstand på 220 Ω til +5 V. Dioderne vil så lyse, når de tilsvarende ben gøres Lave. Find ud af, hvilke ben der svarer til hvilke segmenter, ved at prøve jer frem.
For at man kan koble syvsegmentet til udgangen fra adderen 7483, skal de binære tal fra adderen først oversættes til signaler, som kan kobles til syvsegmentet. Det klares med oversætteren (syvsegment-dekoderen) 7447, der placeres på en 16-bens prøveplade med tilhørende seddel over benfordelingen. Udgangene a,b,c,d,e,f,g forbindes til de tilsvarende indgange på syvsegmentet, og når strømforsyningen er klaret, kan man koble indgangene A,B,C,D på oversætteren til Høj og Lav afhængigt af, hvilke tal man ønsker udlæst.
Når det virker, kobles indgangene A,B,C,D til 7447 til udgangene fra adderen 7483 , som blev benyttet i øvelse 1. Prøv, om det virker.

Øvelse 3. Half Adder og Full Adder.

I denne øvelse kan du bygge en Half Adder af to 7400 IC'ere, der hver indeholder fire NAND GATES.
Som du har set kan en Half Adder bygges af en EXCLUSIVE-OR og en AND GATE. På figur 85 er vist et tilsvarende kredsløb med to AND GATES, en OR GATE og en Inverter.

Half Adder.



85 Half Adder af GATES.

Byg kredsløbet af NAND GATES ved at udnytte, at AND, OR og Inverteren hver kan opbygges af lutter NAND GATES. Se evt. figur 57-59 side 33.
I kan også udnytte, at EX-OR kan laves af 4 NAND, som vist på figur 68 side 44.

Opbyg også en Full Adder. I kan evt. benytte tre 7400 eller to 7400 og en 7404.

Øvelse 4. ALU'en 74181.

Betegnelsen ALU står for **A**rithmetic **L**ogic **U**nit og er navnet på IC'en 74181. ALU'en kan, som navnet antyder, lave regne- og logikoperationer, afhængigt af hvordan man styrer den. Da IC'en har 24 ben, er der lavet en speciel printplade, hvor man med kontakter kan styre indgangene til IC'en. Udgangene vises på nogle lysdioder. Se figur 86. Til højre på billedet ses kontakterne til styreindgangene.

Først Logik.
Stil styreindgangen "Aritmetik/Logik" på "Logik". Nu kan ALU'en lave logiske operationer på de 4 sæt input og virker som 4 parallelle gates med indgangene A og B. Udgangen har betegnelsen F.

Prøv først at stille indgangene A og B til de 4 muligheder LL, LH, HL, HH. Så kan man på udgangene se sandhedstavlen for den logiske operation, ALU'en udfører.
Hvilken virkning har et skift af styreindgangene S_0,S_1,S_2,S_3?

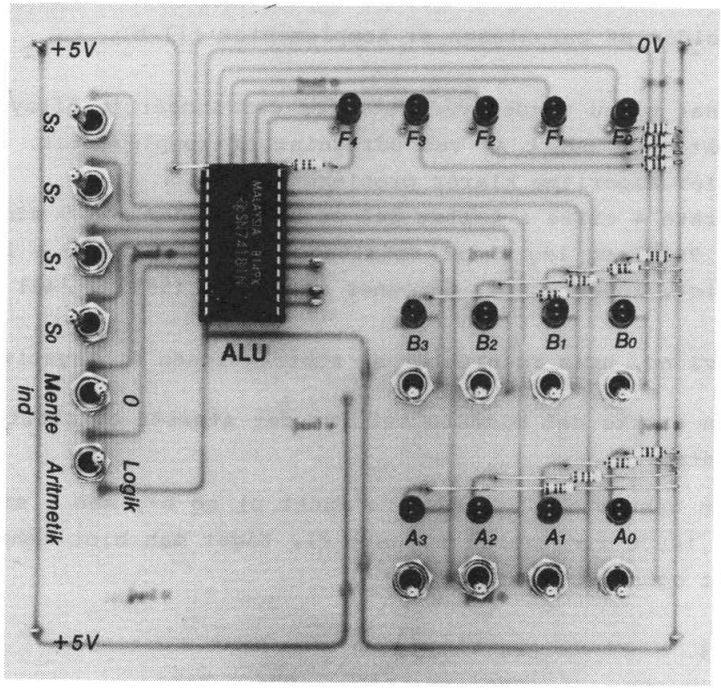
Hvordan skal disse indgange stå, for at vi får udført de logiske operationer AND, OR, NAND, NOR, EX-OR?

Hvorfor kan en ALU udføre netop 16 forskellige logiske operationer?

Dernæst Aritmetik.
Sæt omskifteren på "Aritmetik" og sæt den indgående mente til 0, dvs. Lav. Der er igen forskellige muligheder, som vælges med styreindgangene S_3, S_2, S_1, S_0; der skal nu tages hensyn til menteudgangen F_4.
De vigtigste regneoperationer, ALU'en kan udføre, er vist i tabellen:

"Navn"	S_3 S_2 S_1 S_0	Virkning
ADD A,B	H L L H	Addition af A og B
DEC A	H H H H	A minus 1
SH L A	H H L L	Skift A en plads til venstre (A+A)
A	L L L L	A

Indgangen "Mente Ind" benyttes, når flere ALU'er kobles efter hinanden. Så skal menteudgangen fra den første kobles til menteindgangen for den næste. Styreindgangene skal i denne situation kobles parallelt.



86 ALU'en 74181, der kan lave regne- og logikoperationer. Funktionen vælges med styrekontakterne til venstre på printkortet. Udlæsningen sker på lysdioder.

Supplerende stof

Subtraktion og negative tal.
For at undgå at subtrahere i en computer omformes de negative tal, så man kan klare en subtraktion ved i stedet at udføre en addition. For at forstå, hvordan computeren behandler negative tal i to-tal-systemet, ser vi først på den tilsvarende metode i ti-talsystemet. F.eks. kan subtraktionen 875 - 344 foregå i følgende trin:

a) -344 --> 10000 - 344 = 9656
b) 875 + 9656 = 10531
c) 10531 --> 0531 --> 531

Altså: i stedet for at trække 344 fra, lægges "ti-komplementet" 9656 til, og det mest betydende ciffer i resultatet kastes væk.
Tallet 10000 i eksemplet er blot valgt, så det er større end 875.
At metoden altid giver det rigtige resultat ses af omskrivningen:

a - b = a + (10000 - b) - 10000

hvor indholdet af parentesen er komplementet til b.

Men hvad har vi nu vundet ved det, for der indgår jo tilsyneladende en subtraktion i punkt a) ved udregning af komplementet. Følgende metode eller algoritme klarer problemet for os :
Tag de første 4 cifre i tallet 344 og udskift dem med 9 minus cifret (0344 --> 9655) og læg 1 til resultatet. 0344 --> 9655 + 1 = 9656.
Komplementet, 10000 - 344, udregnes altså som (9999 - 344) + 1.

Hvad gør vi nu, hvis resultatet af subtraktionen er negativt?

1. Man kan trække det mindste tal fra det største og skifte fortegn på resultatet.

2. Man kan gennemføre procedurens punkt a) og b), men i stedet for at trække 10000 fra tallet i punkt c), tager man blot komplementet til tallet og skifter fortegn.

Eksempel 3. 344 - 875
a) -875 --> 10000 - 875 = 9125
b) 344 + 9125 = 9469
c) 9469 --> 0531 --> -531

Opgave 10. Udregn ved komplementmetoden

a) 3700 - 1744 b) 100 - 64 c) 222 - 444

Nu til to-talsystemet!

Lad os antage, at vores computer regner med 8 bit ad gangen. Vi vælger nu den venstre bit til fortegnsbit, så en fortegnsbit på 0 svarer til et positivt tal, og en fortegnsbit på 1 svarer til et negativt tal. Det største tal computeren kan rumme, er tallet 01111111 = + 127. Ved de negative tal angiver vi i stedet for tallet to-komplementet til tallet. to-komplementet fremkommer efter følgende opskrift: skift alle nuller ud med et'taller og omvendt (inklusiv fortegnsbitten) og læg 1 til resultatet.
Eksempel, hvor 7 omformes til -7:

00000111 --> **1**1111000 + 1 = **1**1111001

Tallet -7 repræsenteres ved bitmønstret 11111001 - altså ved komplementet. Dermed er fortegnsbitten altid 1 for negative tal.
På denne måde bliver negative tal fra -1 til -127 repræsenteret ved deres komplementer.

Subtraktion af tal klares nu ved at addere det tilsvarende negative tal (komplementet), som vi skal se:

Eksempel 4. 32 - 19

00100000 - **0**0010011 =	32 - 19
00100000 + **1**1101101 =	32 + (-19)
00001101	13

Bemærk, at menten fra additionen (bit nr 9) forsvinder af sig selv, da vi jo kun regner med 8 bit.

Opgave 11. Udregn 115 - 35 og 76 - 112 i to-talsystemet.
Hvordan repræsenteres tallene -16, -100, -127 og -0 i computeren?
Vis, at komplementet til et positivt tal i computeren er negativt og omvendt.

Opgave 12. I sproget "Poly-Pascal" går de hele tal fra -32768 til +32767, og andre hele tal eksisterer ikke. Hvor mange bit må man afsætte plads til pr. helt tal i dette sprog?

Division.

På samme måde som vi kan gange i to-talsystemet, kan vi også dividere. I det følgende holder vi os til heltalsdivision, da det er det simpleste, men metoderne kan udvides til også at gælde kommatal. Når vi i ti-talsystemet dividerer 33 med 4, tæller vi, hvor mange gange tallet 4 kan trækkes fra 33, uden at resultatet bliver negativt. 33 : 4 = 8 og rest 1. Vi kan gøre prøve $33 = 4 \cdot 8 + 1$.
For at gøre det simpelt kan vi prøve os frem ved at undersøge, om $1 \cdot 4$ er støre end 33 , om $2 \cdot 4$ er større end 33 , om $3 \cdot 4$ er større end 33 osv. Divisionen er slut, når vi har det største tal x, så $x \cdot 4$ er mindre end 33. Resultatet af divisionen er 8 og resten findes som $33 - 8 \cdot 4 = 1$.

Dette kan vi også gøre i to-talsystemet

$33_{10} = 100001_2$ og $4_{10} = 100_2$.

$1 \cdot 100 = 100$
$10 \cdot 100 = 1000$
$11 \cdot 100 = 1100$
$100 \cdot 100 = 10000$
$101 \cdot 100 = 10100$
$110 \cdot 100 = 11000$
$111 \cdot 100 = 11100$
$100'0 \cdot 100 = 100000$
$1001 \cdot 100 = 100100$

Som det fremgår af udregningerne, er resultatet af divisionen $1000_2 = 8_{10}$, da det næste resultat er større end $33_{10} = 100001_2$.
Denne metode kan ligesom i ti-talsystemet gøres smartere ved at udnytte positionssystemet.

Eksempelvis kan vi udføre divisionen 57 : 5 i to-talsystemet:
$57_{10} = 111001_2$ og $5_{10} = 101_2$.

```
111001 : 101 = 1011 og rest 10
101
 100
 000
 1000
  101
   111
   101
    10
```

Hvad er i øvrigt resultatet af divisionen?

Bemærk, at når tallene skal subtraheres, sker det i computeren ved at komplementerne til tallene adderes.

Opgave 13. Udfør følgende divisioner i to-talsystemet:

a) 44 : 7 b) 32 : 4 c) 100 : 10

Store og små tal.

Ligesom på en lommeregner har man i en computer mulighed for at regne med ti-potenser, når meget store og meget små tal skal angives. Derfor må man i computeren afsætte plads til både tallets cifre og ti-potensen, foruden at der skal være plads til fortegn på både cifrene og eksponenten. Sædvanligvis angives i stedet for ti-potensen en to-potens, da computere jo for det meste regner i to-talsystemet. Lad os sige, at vi i vores computer afsætter 2 gange 8 bit til hvert tal. I de første 8 bit angives tallets cifre med fortegn, og i de sidste 8 bit angiver vi eksponenten med fortegn.

Eksempel 6. Tallet $+1{,}110100 \cdot 2^{+0001011}$ skrives med to bytes, hvor en byte blot er 8 bit:

01,110100 **0**0001011

Kommaets placering i den første byte er underforstået.

Opgave 14. Vis, at tallet i eksemplet ovenfor er $3{,}721 \cdot 10^3$.
Hvad er det største tal, vi kan angive i vores tænkte computer?

Opgave 15. Hvad er det mindste positive tal, vi kan angive i vores tænkte computer, og hvordan angives det? Hvad er det mindste tal?

Opgave 16. I sproget "Pascal" er grænserne for de reelle positive tal $+1 \cdot 10^{38}$ og $+1 \cdot 10^{-38}$, og desuden er der mulighed for 11 betydende cifre, når tallene oversættes til ti-talsystemet.
I sproget "Coma180" er de tilsvarende grænser for positive tal $1{,}000000000000 \cdot 10^{-128}$ til $9{,}999999999999 \cdot 10^{126}$. Hvor mange bit skal der afsættes plads til pr. tal ved de to sprog?

Hvor mange forskellige tal er der i de 2 sprog?

Hvad er afstanden mellem det største og det næststørste tal?

Omsætning af tal fra to- til ti-talsystemet.

Vi vil i det følgende nøjes med at se på omsætning af hele tal. Et tal skrevet i to-talsystemet består af et antal cifre. Første ciffer fra højre, det mindst betydende ciffer, angiver antallet af enere i tallet, og de næste cifre angiver antallet af toere, firere, ottere osv.

F.eks. er tallet $110010_2 = 1 \cdot 32 + 1 \cdot 16 + 0 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 0 \cdot 1 = 50$.

Denne metode eller algoritme til omsætning af binære tal til ti-talsystemet er meget anvendelig ved hovedregning, da udregningerne foregår i ti-talsystemet. Men det er en meget besværlig metode at administrere i en computer, og derfor benytter man her algoritmen, der omtales i det følgende.

Metoden bygger på det gamle princip med at tælle ved at ordne tallet i bundter af ti, "tibundtningsprincippet". Tibundterne kan igen bundtes i ti til "hundredebundter" osv. Når tallet angives, angiver vi først antallet af enere, dernæst antallet af tibundter, af hundredebundter osv. F.eks. vil tallet 835 i første omgang blive talt som 5 enere og 83 tibundter. Dernæst bundtes tierne i bundter af hundredebundter, og tallet angives nu som 5 enere, 3 tiere og 8 hundreder.

Først koncentrerer vi os om antallet af enere i tallet, der skal skrives i ti-talsystemet. Dertil deler vi først tallet med ti og finder antallet af tibundter. Resten må være antallet af enere i tallet. Altså første ciffer fra højre.

Hvis vi kalder tallet, der skal omsættes, for x, får vi

$x : ti = y_0$ og rest r_0. (835 : ti = 83 og rest 5)

Resten r_0 er antallet af enere i tallet og dermed første ciffer.

Hvis nu y_0, der er resultatet af divisionen, ligger mellem 0 og 9, er vi færdige, for da er y_0 antallet af tiere i tallet x, og tallet skrives som $y_0 r_0$ i ti-talsystemet

Men hvis y_0 er et tal større end 9, er vores oprindelige tal x på mere end 2 cifre, når det angives i ti-talsystmet. Vi skal for at finde næste ciffer i tallet x finde antallet af tiere i tallet. Men nu gælder det blot om at opløse tallet y_0 i et antal enere (antallet af tiere i x) og et antal tiere (antallet af hundrede i x) osv. Vi er altså i nøjagtig samme situation, som før - blot med tallet y_0 i stedet for x. Derfor findes andet ciffer af tallet x ved at udføre

divisionen:

y_0 : ti = y_1 og rest r_1. (83 : ti = 8 og rest 3)

Da tallet r_1 er antallet af enere i tallet y_0, er det dermed også antallet af tiere i tallet x. Altså andet ciffer i tallet x.

Resten af cifrene findes på nøjagtig samme måde - ved at benytte samme algoritme på tallene y_0, y_1, y_2 osv. Og selv om denne metode ser mere kompliceret ud end den første, har den sidste metode den fordel, at den kun benytter sig af udregninger i to-talsystemet, så metoden er velegnet til en computer. Husk her, at ti = 1010_2. Desuden er det samme metode ved alle tallets cifre, så man skal blot lave et program, der kan finde første ciffer. Derefter kan computeren genbruge programmet ved udregning af de resterende cifre.

Eksempel 7.

Vi ønsker at omsætte det binære tal 11110011 til ti-talsystemet.

a) 11110011_2 : 1010_2 = 11000_2 og resten 11_2 (tre)

b) 11000_2 : 1010_2 = 10_2 og resten 100_2 (fire)

c) 10_2 : 1010_2 = 0_2 og resten 10_2 (to)

Dermed er tallet 11110011_2 = 243_{10}

Opgave 17. Check, om udregningerne i eksemplet er rigtige.

Opgave 18. Omsæt følgende binære tal til ti-talsystemet.

a) 1101 b) 11111111 c) 100000000 d) 1100100

Algoritmen med at dividere kan let overføres til andre talsystemer, idet tallet 10 blot udskiftes med et andet tal. Fastsættes tallet til 2, omsættes tallene til to-talsystemet.

Omsætning fra ti- til to-talsystemet.

Eksempel 8.

Tallet 55 omsættes til to-talsystemet på følgende måde :

a) 55 : 2 = 27 og rest 1 (laveste ciffer er 1)

b) 27 : 2 = 13 1

c) 13 : 2 = 6 1

d) 6 : 2 = 3 0

e) 3 : 2 = 1 1

f) 1 : 2 = 0 1

Dermed er tallet $55_{10} = 110111_2$

Bemærk, at udregningerne foregår i det talsystem, vi bevæger os fra - altså her i ti-talsystemet.

Opgave 19. Omsæt følgende ti-tal til både to- og fire-talsystemet:

a) 129 b) 31 c) 100 d) 3299

ASCII-koder.

Når computeren omsætter tal fra to- til ti-talsystemet, foregår udregningerne, som vi har set, i to-talsystemet, men computeren slipper ikke for at skulle benytte cifrene fra 0 til 9, når tallene er omsat til ti-talsystemet, og resultaterne skal sendes til skærmen eller printeren. Det sker ved, at tallene sendes til skærm eller printer som binære koder, hvor den mest benyttede kode er ASCII-koden eller afarter deraf. ASCII står for American Standard Code for Information Interchange. Det er en kode, hvor symbolerne, dvs. tal og bogstaver og en række styrekarakterer, sendes som 8 bit, hvor de 7 første er den egentlige kode og hvor den 8. bit benyttes til kontrol. En tabel over ASCII-koden findes i enhver computermanual med de ændringer, det altid er nødvendigt at foretage, for at nan ikke skal nøjes med det amerikanske tegnsæt. Tallene fra 0 til 9 angives i ASCII-koden ved den binære kode 00110000 + "cifret binært". Det betyder, at computeren blot skal addere tallet 00110000 til cifrene, før de sendes til de elektroniske enheder, der styrer printeren eller skærmen.

BCD-koder.

For helt at slippe for at skulle omsætte tal mellem forskellige talsystemer, benytter man ved de fleste lommeregnere og enkelte computere, der hovedsageligt benyttes ved regnskaber og økonomiske transaktioner, en anden teknik. I stedet for at regne i to-talsystemet, afsættes 4 bit pr. ciffer i ti-talsystemet, og cifrene fra 0 til 9 oversættes nu til de binære koder fra 0000 til 1001. Disse koder betegnes BCD, Binary Coded Decimal, og tallene opfattes stadigt i ti-talsystemet. Derfor må alle udregninger foregå i ti-talsystemet, og det gør dem noget mere kompliceret. Men da de fleste CPU'ere er konstrueret til at udføre BCD-regneoperationer, er problemet overkommeligt. Regneenheden i en lommeregner er ligefrem bygget til at regne i BCD. Når resultaterne skal ud af maskinen, er det blot at sende hver ciffer til hvert sit segment på lommeregnerens display eller som ASCII-kode til skærmen, og man undgår at skulle omsætte tal mellem forskellige talsystemer.

4. Hukommelse

Oversigt

Regnelageret i en computer gennemgås i dette kapitel. Først ser vi på kommunikationen mellem regnelageret og computerens regneenhed, og siden beskrives en simpel model af et regnelager.
Øvelserne indeholder foruden bygningen af hukommelsesenheder og sammensætningen og afprøvningen af disse også afprøvningen af den færdige hukommelses-IC 2112.
I det supplerende stof kan man læse om andre typer hukommelser og om, hvordan man udvider et regnelager fra en enkelt til flere IC'er.

Hovedtekst

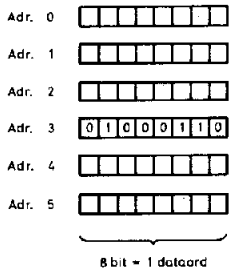
Den elektroniske hukommelse er en meget vigtig del af en computer. Det er hukommelsen, som regneenheden kommunikerer med, når den udfører et program. Såvel instruktionerne til regneenheden som de data, regneenheden skal manipulere med, er placeret i den elektroniske hukommelse. En sådan hukommelse kaldes et regnelager eller en RAM, og det er sådanne hukommelser, vi vil undersøge i dette kapitel.
For at forstå opbygningen af et regnelager vil vi se på, hvordan kommunikationen foregår mellem regneenheden og hukommelsen. Kommunikationen sker gennem to sæt ledninger: adresseledningerne kaldet adressebussen og dataledningerne kaldet databussen. Desuden er der et par ledninger til kontrol af hukommelsen.

Adressebussen.
Adressebussen består af et antal parallelle ledninger fra regneenheden til hukommelsen. Med disse ledninger styrer regneenheden, hvor i hukommelsen kommunikationen skal foregå. Hukommelsen er nemlig delt op i et antal positioner, der hver har en adresse. Til sådan en adresse svarer en ganske bestemt kombination af spændinger på adresseledningerne, og regneenheden kan derfor kun kommunikere med en adresse ad gangen. Med 8 ledninger i adressebussen er det muligt at nå 2^8 = 256 adresser i hukommelsen. Og med 16 ledninger i adressebussen er det muligt at nå 2^{16} = 65536 = 64 K adresser.

Databussen.

Databussen består ligeledes af et antal parallelle ledninger mellem regneenheden og hukommelsen, i reglen 8, 16 eller 32 ledninger. Det er gennem databussen, informationerne sendes. Er der 16 ledninger i databussen, flyttes 16 bit til eller fra hukommelsen ad gangen, og man siger, at ordlængden er på 16 bit. Ordlængden er altså bestemt af antallet af ledninger i databussen, og da det netop er et dataord, der flyttes ad gangen, er hukommelsen indrettet, så hver adresse netop kan lagre et dataord. Er der f.eks. 256 adresser med 16 bit i hver, siges hukommelsen at have kapaciteten 256 · 16 bit.

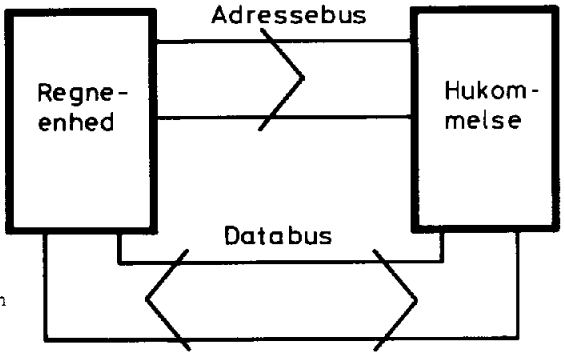
En hukommelse kan altså opfattes som et antal enheder, der kan indeholde en bit. Enhederne er samlet i grupper på 8, 16 eller 32, der hver har fået et adresse-nummer. På figur 87 er vist opbygningen af en hukommelse med en ordlængde på 8 bit og med 6 adresser. Kapaciteten er altså på 6 · 8 bit.



87 Gruppering af hukommelsesenheder.

Kommunikation med hukommelsen.

På figur 88 er vist, hvordan regneenheden og hukommelsen er forbundet via adressebussen og databussen. Pilene indikerer, at signalerne i adressebussen sendes fra regneenheden til hukommelsen, mens de i databussen kan gå begge veje.



88 Kommunikation mellem regneenhed og hukommelse sker gennem databussen og styres med adressebussen.

Til hver hukommelses-IC skal der være mindst to kontrolledninger. En til at styre, om der skal læses ud eller skrives ind i hukommelsen, kaldet R/W for Read/Write. En anden til at styre, hvornår udlæsningen/indskrivningen skal foregå, kaldet CS for Chip Select.

Skrive ind i hukommelsen.
Skal f.eks. dataordet 00111001 skrives ind i hukommelsens adresse 3, sættes adresseledningerne i adressebussen til LLLLLLHH = 3. Dataledningerne i databussen sættes til dataordet LLHHHLLH = 00111001, og kontrolledningen R/W gøres Lav for at skrive ind. Mens alle disse ledninger sættes til de rigtige spændinger, er kontrolledningen CS Høj, og først når alle spændingerne er stabile, og indskrivningen skal foregå, gøres kontrolledningen CS Lav et øjeblik. I løbet af 10^{-8} sek er dataordet lagret i adresse 3, hvor det huskes, indtil et nyt dataord skrives oveni eller indtil strømmen afbrydes.

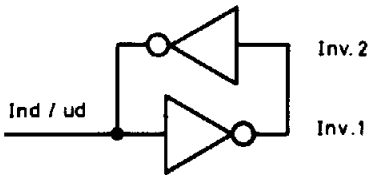
Læse ud af hukommelsen.
Skal f.eks. informationen, der er gemt i adresse 15, læses af regneenheden, sættes først adresseledningerne i adressebussen til LLLLHHHH = 15, og kontrolledningen R/W gøres Høj svarende til udlæsning. Når regneenheden er klar til at modtage informationen, gøres kontrolledningen CS Lav, og informationen i adresse 15 kan nu læses på dataledningerne. Efter en udlæsning er indholdet af hukommelsen stadigt det samme.
Bemærk, at databussen flytter informationerne fra regneenheden til hukommelsen, når der skrives ind, og modsat, når der læses ud. Derimod styres adressebussen kun af regneenheden.

Opbygning af en hukommelse.

Hukommelsesenhederne i computeres regnelagre har i tidens løb været lavet på mange forskellige måder svarende til den billigste måde at producere dem på. De første store regnelagre var opbygget af magnetiske ringe, som enten kunne være magnetiseret den ene eller den anden vej svarende til Høj og Lav. I øjeblikket produceres store hukommelses-IC'er billigst ved at oplade eller aflade meget små områder af en IC for hver bit. Det er de såkaldte dynamiske hukommelser, hvor der kan være op imod 10^5 hukommelsesenheder i en enkelt IC. De har dog den store ulempe, at de glemmer al informationen i løbet af 1/1000 sek og derfor hele tiden må genopfriskes. Vi holder os til de noget dyrere og mere skikkelige hukommelser, der husker,

så længe der er strøm - de statiske hukommelser.
I det følgende bygges en model af en hukommelses-IC, og vi har valgt at bygge den så enkel som muligt.

Hukommelsesenheden.
Hukommelsesenheden er her bygget af to invertere, og som vist på figur 89 er udgangen af inverter 1 forbundet til indgangen af inverter 2 og omvendt. Se evt. øvelse 9 side 46 i kap. 2.



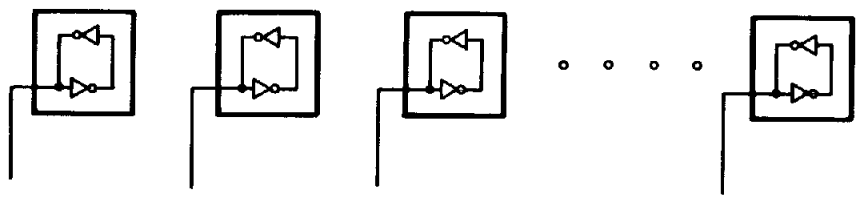
89 Hukommelsesenhed af 2 Invertere.

Når hukommelsesenheden skal huske et H, tvinges ledningen til hukommelsesenheden blot Høj. Øjeblikkeligt derefter bliver udgangen på inverter 1 Lav, og da denne udgang er forbundet til indgangen af inverter 2, bliver dennes udgang Høj. Nu er situationen stabil, og de to invertere vil styre hinanden, til strømmen afbrydes.

Skal hukommelsen huske Lav, tvinges enheden blot Lav, og 10^{-8} sek senere er informationen husket. Situationen er igen stabil. Overvej!

Udlæsningen af hukommelsesenheden kan foregå ved at undersøge spændingen på hukommelsesenheden, dvs. inverter 2's udgang. Man kan f.eks. benytte et udlæsemodul.

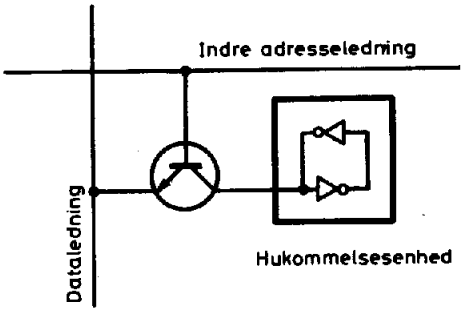
Gruppering af hukommelsesenheder.
Hvis computeren er indrettet, så den regner med 8 bit ad gangen og altså har en ordlængde på 8 bit, skal hukommelsen kunne lagre 8 bit i en adresse. På figur 90 er vist en gruppering af 8 hukommelsesenheder til lagring af et dataord.



90 8 hukommelsesenheder i én adresse.

Som figuren viser, skal der være 8 parallelle ledninger til hukommelsesenhederne. Disse udgør databussen.
I store hukommelses-IC'er kan der gemmes flere tusinde dataord ad gangen, og vi skal nu se, hvordan man elektronisk kommer i kontakt med en enkelt adresse.

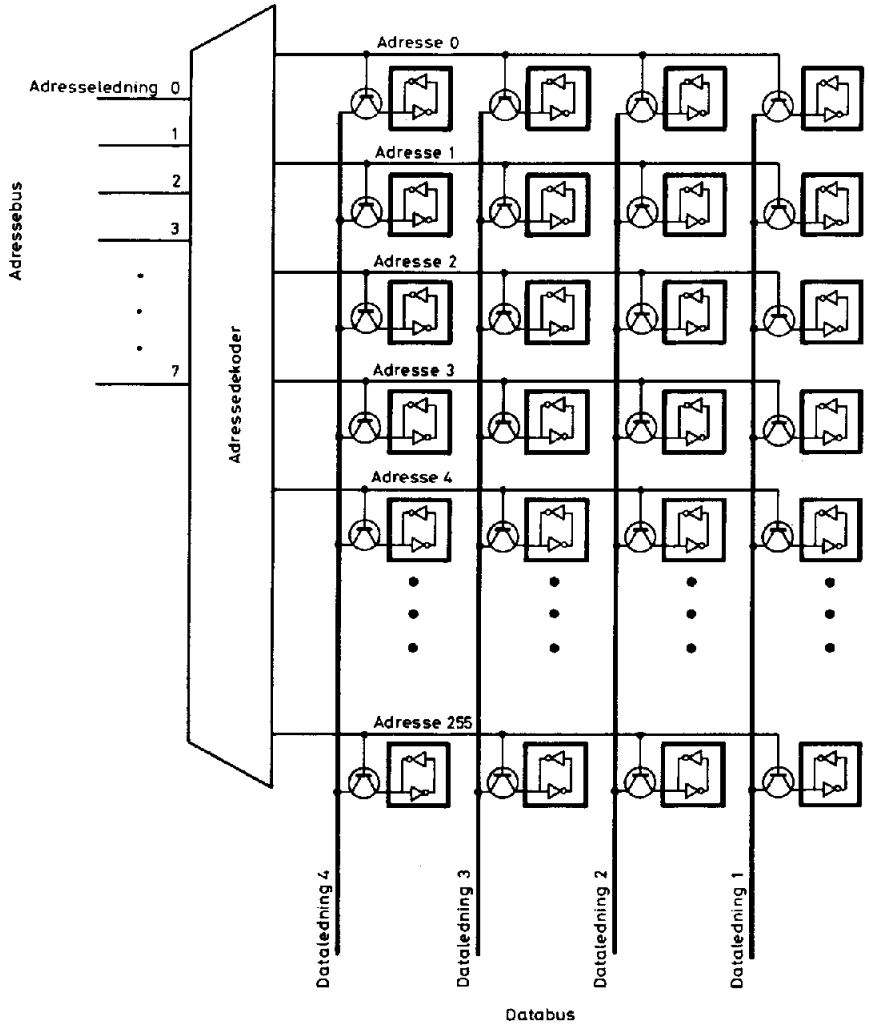
Elektronisk adskillelse af de forskellige adresser.
Vores hukommelsesenhed kobles via en transistor til dataledningen. Transistoren fungerer som en styret kontakt og har til opgave at koble hukommelsesenheden til og fra dataledningen. Transistoren er vist sammen med hukommelsesenheden på figur 91.



91 Hukommelsesenhederne kobles til og fra dataledningen med en transistor og den indre adresseledning.

For at man kan komme i kontakt med hver af de mange adresser, er der en ledning for hver adresse, tegnet vandret. Vi kalder den for en indre adresseledning for ikke at forveksle den med ledningerne i adressebussen. Nu kan vi koble en adresse til eller fra dataledningerne ved at gøre den tilhørende indre adresseledning Høj eller Lav. Er ledningen Høj, leder transistoren, og enhederne er koblet til dataledningerne. Forbindelserne er afbrudt, når adresseledningen er Lav. Da man kun kan kommunikere med en adresse ad gangen, må man sørge for, at netop en indre adresseledning er Høj ad gangen.
Man kunne i princippet lade alle de indre adresseledninger udgøre adressebussen, men med en hukommelse på 256 K skulle der være 262144 = 256 K ledninger i adressebussen og et tilsvarende antal ben på IC'erne. Derfor sender man adressenummeret binært via adressebussen. Nu svarer en ganske bestemt kombination af spændinger på adressebussen til netop en adresse, og det er nok med 18 adresseledninger i adressebussen, idet 2^{18} = 256 K. Inden i hukommelses-IC'en sidder der en dekoder, der aflæser nummeret på adressebussen og derefter gør den indre adresseledning med det rigtige nummer Høj.

Ved opbygningen af computeren benyttes hukommelses-IC'en 2112, og derfor er 2112 vist skematisk på figur 92. Som andre hukommelser, der både kan læses ud af og skrives ind i, har 2112 en indgang til at styre indskrivningen og en indgang til at aktivere hukommelsen. For overskuelighedens skyld er de ikke vist på figuren.
2112 er en statisk hukommelse med en kapacitet på 256 · 4 bit, og derfor er der 4 dataledninger og 8 adresseledninger.



92 Skematisk opbygning af hukommelses-IC'en 2112.

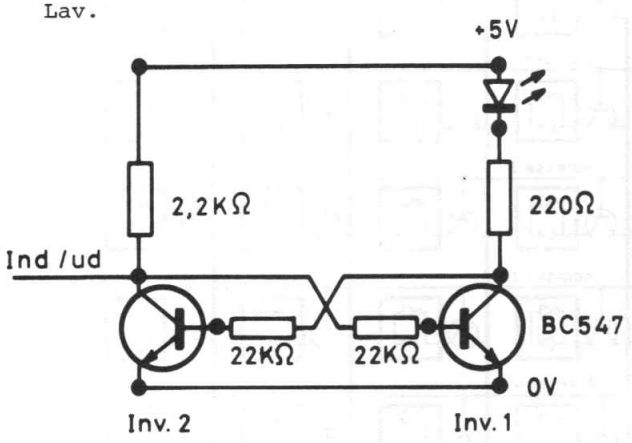
Øvelser

Hukommelsesenheder.

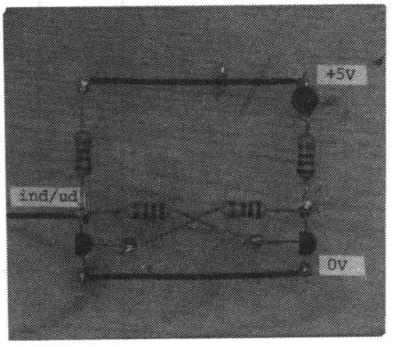
Som vi har set det i kapitel 1, kan inverteren bygges af en transistor og to modstande. I øvelse 1a bygges hukommelsesenheden på et sømbræt af disse komponenter. I øvelse 1b benyttes færdige invertere fra IC'en 7404.

Øvelse 1a. Hukommelsesenhed på sømbræt.

To invertere kobles sammen på samme sømbræt efter diagrammet på figur 93. Et billede af den færdige enhed er vist på figur 94. Den ekstra lysdiode er medtaget, for at man kan se indholdet af hukommelsesenheden. Bemærk, at dioden lyser, når inverter 1's udgang er Lav.



93 Hukommelsesenhed



94 Hukommelsesenhed på sømbræt.

Afprøv hukommelsen ved at koble et udlæsemodul til hukommelsesenhedens ind/udgang, og prøv at gøre denne ledning Høj og Lav.

Øvelse 1b. Hukommelsesenhed med IC'en 7404.

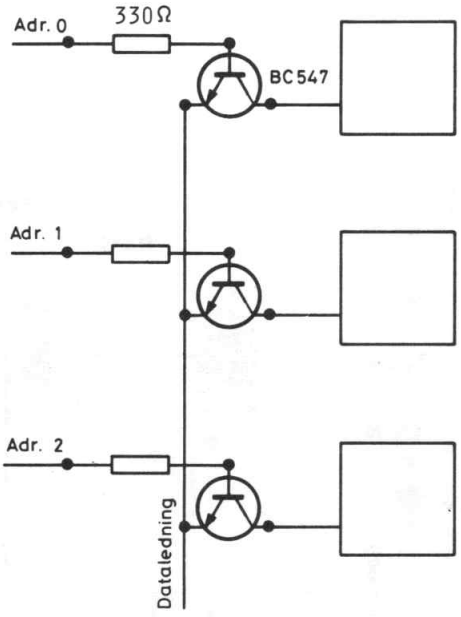
IC'en 7404 med 6 stk. NOT GATE monteres på en prøveplade, og en seddel over benfordelingen lægges på. Lav 3 hukommelsesenheder på pladen.

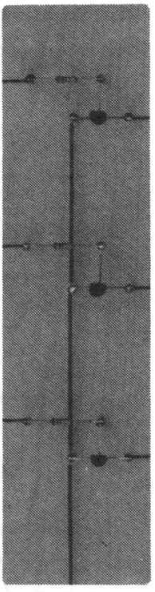
Indskrivning foregår ved, at ledningen til en af enhederne tvinges Høj eller Lav. Om indskrivningen er foregået rigtigt, kan man først undersøge, når man læser ud.

Udlæsningen foregår ved, at ledningerne til de 3 enheder forbindes til et udlæsemodul.

Øvelse 2. Dataledning og adresser.

3 hukommelsesenheder kobles til den samme dataledning på følgende opstilling, der bygges af 3 transistorer og 3 modstande.





95 Dataledning til hukommelsesenheder

96 Dataledning på sømbræt.

Forbind 3 hukommelsesenheder til dataledningen som vist på figuren. Både enheder fra øvelse 1a og 1b kan benyttes.

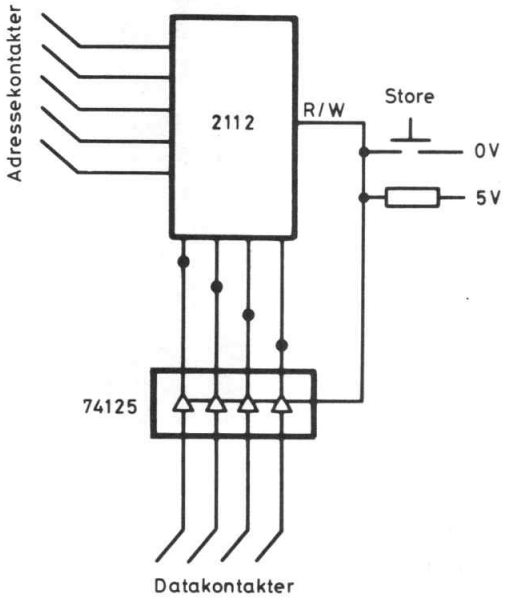
Udlæsning: Vil man undersøge indholdet af adresse 2, gøres ledningen til denne adresse Høj. Transistoren slutter nu forbindelsen mellem enheden og dataledningen, og sandhedsværdien kan aflæses på udlæsemodulet tilsluttet dataledningen.

Indskrivning: Ledningen til adresse 1 gøres Høj, og dataledningen tvinges til enten Høj eller Lav. Indskrivningen er så klaret i adresse 1.

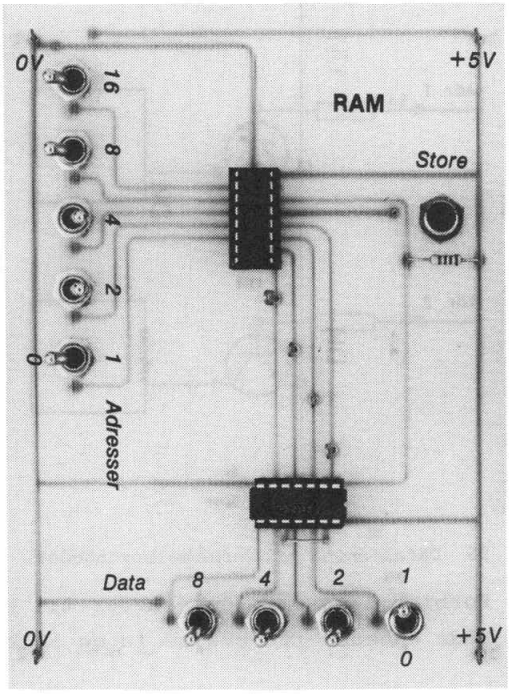
Denne opstilling giver en hukommelse med en kapacitet på 3 · 1 bit.
Prøv at bygge en 3 · 2 og en 6 · 1 bits hukommelse.
Den første kan laves med to enheder koblet, så dataledningerne sidder parallelt. Den sidste laves ved, at man kobler dataledningerne i serie efter hinanden.

Øvelse 3. Den færdige RAM 2112.

IC'en 2112 er en af de simpleste hukommelser, man kan købe, med en kapacitet på 256 · 4 bit. Afprøvningen foregår på en færdig printplade sammen med den tilhørende IC 74125. Hukommelsen har 8 adresseben og 4 databen, og for at gøre det simpelt, benytter vi kun 5 af adressebenene. Et diagram over hukommelsen er vist på figur 97, og et billede af den tilhørende printplade er vist ved siden af.



97 Diagram af RAM-kort



98 Billede af RAM-kort

Hukommelsen er forbundet med adresse- og datakontakter, så man let kan styre adresser og data. IC'en 74125 (tri-state buffere) sørger for, at datakontakterne kun kobles til hukommelsen, når der skrives ind, og dens styreindgang er derfor koblet sammen med kontrolledningen R/W til RAM'en.

Udlæsning: Adressekontakterne sættes til den ønskede adresse i to-talsystemet, og de lagrede data kan aflæses på dataledningerne, når de 4 printspyd på databussen forbindes til hver sin lysdiode på et udlæsemodul.

Indskrivningen finder sted ved, at man sætter både adressekontakter og datakontakter til det ønskede og derefter trykker på "store-knappen".

Supplerende stof

Forskellige typer hukommelser.

RAM står for Random Access Memory, hvilket henviser til, at man kan komme i forbindelse med hukommelsens adresser i vilkårlig rækkefølge. Dette i modsætning til et magnetisk bånd, hvor informationerne er lagret i en bestemt rækkefølge. RAM'en kaldes også for en læse/-skrive-hukommelse, da man både kan skrive informationer ind og læse dem ud af hukommelsen. En RAM benyttes altid i forbindelse med regnelageret i en computer. Undertiden kaldes regnelageret og regne-enheden under et for CPU = Central Processing Unit. Når der er tale om IC'ere, er det dog kun regne- og styreenheden, der benævnes CPU.

ROM står for Read Only Memory og betegner en type hukommelse, man kun kan læse ud af, og som derfor ikke kan slettes. En ROM er programmeret en gang for alle af fabrikanten. ROM'en benyttes som regel i en computer til lagring af styresystemer, oversættere og andre standardprogrammer. En ROM kan dog også fungere som en over-sætter af elektriske signaler med indgang på adresseledningerne og udgang fra dataledningerne. Så skal der i hver adresse være lagret den kombination af spændinger, der svarer til udgangssignalerne. Adressenummeret svarer til kombinationen af indgangsspændinger.

PROM står for Programmable ROM og betegner en speciel type ROM, som man selv kan programmere eller lægge data ind i. Laver man fejl under programmeringen, kastes PROM'en bort, og man starter forfra.

EPROM står for Erasable PROM og betegner en speciel type PROM, hvor man selv kan slette indholdet. Det sker som regel med ultraviolet lys gennem et særligt vindue i IC-en. Dermed slettes al information på en gang. Dette i modsætning til RAM'en, hvor man sletter indhol-det af en adresse ad gangen.

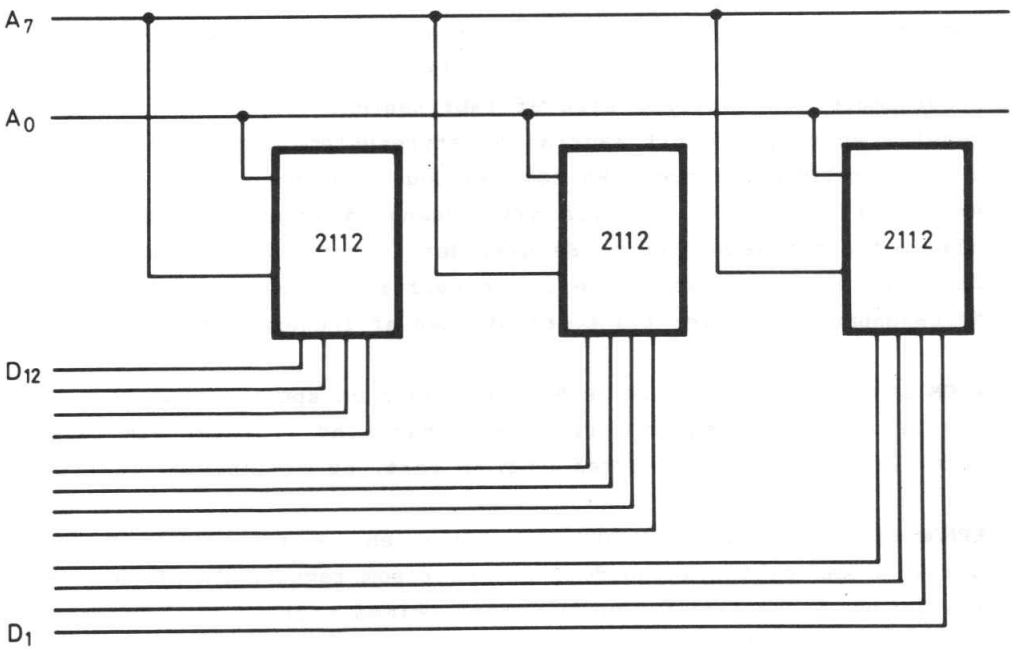
EEROM står for Electrical Erasable ROM og betegner en ny type EPROM, hvor man kan slette en adresse ad gangen med et specielt elektrisk signal. Så bortset fra, at sletningen er langsom, ca. 50 msek pr. adresse, fungerer en EEROM som en RAM. EEROM'en har dog den fordel, at indholdet huskes - også efter at strømmen er afbrudt.

Udvidelse af regnelageret.

Ofte er der ikke kapacitet nok i en enkelt IC som regnelager i en computer, og derfor kobles mange ens IC'ere sammen til et større regnelager. Udvidelsen kan foregå på to principielt forskellige måder. Enten kan man udvide antallet af ledninger i databussen, eller man kan udvide antallet af ledninger i adressebussen.

Udvidelse af databussen.

På figur 99 er vist et eksempel, hvor 3 stk. 2112, hver med en kapacitet på 256 · 4 bit, er koblet sammen til en hukommelse med kapaciteten 256 · 12 bit.



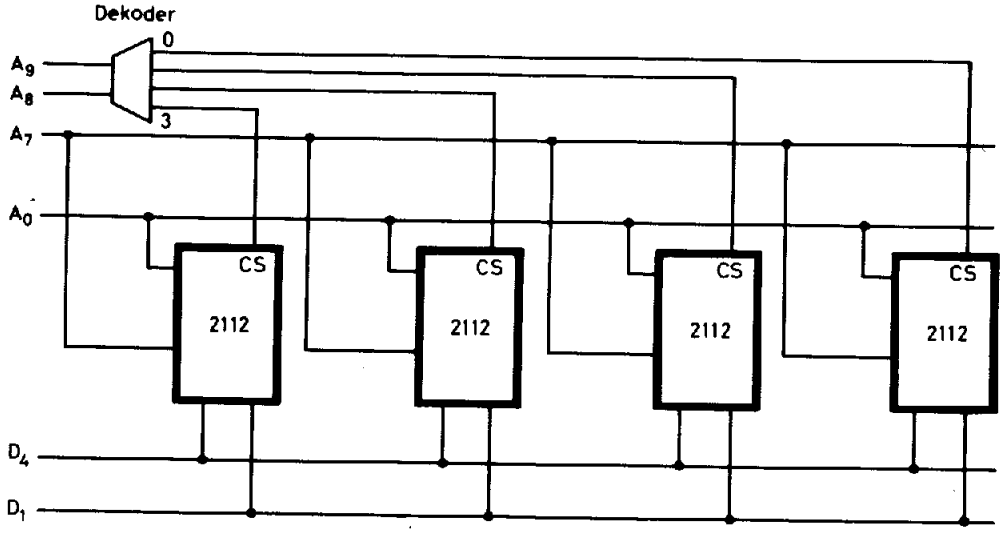
99 Sammenkobling af 3 stk. 2112 til et regnelager på 256 · 12 bit.

Opgave 1. Informationen på følgende 12 bit, HHHLLLLLHLHL, er gemt i adresse 44. Hvilken information er gemt i hver af de 3 IC'ere?

Opgave 2. Hvad er den største ordlængde, man kan have, med 8 stk. 2112?

Udvidelse af adressebussen.

På figur 100 er vist et eksempel, hvor 4 stk. 2112 er forbundet til en hukommelse med en samlet kapacitet på 1024 · 4 bit.



100 Sammenkobling af 4 stk. 2112 til et regnelager på 1024 · 4 bit.

Opgave 3. Informationen HHLH er gemt i den samlede adresse 508. I hvilken IC er informationen gemt og i hvilken adresse heri?

Opgave 4. Hvor mange stk 2112 skal man bruge, for at lave en hukommelse med en kapacitet på 64 K · 8 bit og hvordan skal de forbindes? Udregn prisen for et sådant statisk regnelager, når en enkelt 2112 koster ca. 30 kr i 1985!

Opgave 5. Hvor stort et lager skal man lave, for at det kan rumme alle personnumre i Danmark?

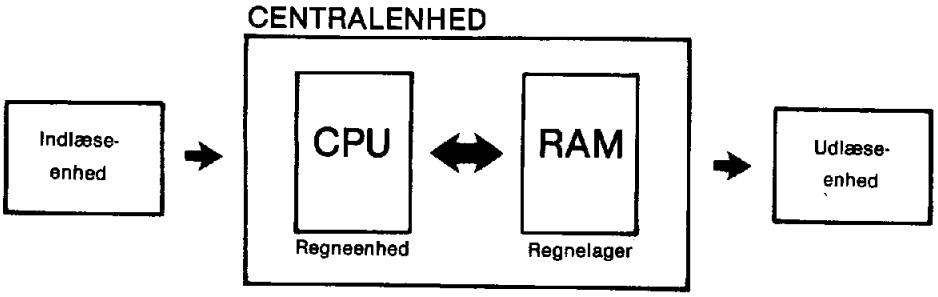
5. Computeren

Oversigt

Hovedteksten indledes med en generel omtale af computeres opbygning, og dernæst beskrives computermodellen, der anvendes i øvelserne, mere detaljeret. Den indeholder en Z80 mikroprocessor, som også omtales. Øvelserne indeholder små programeksempler til computermodellen, hvor de simpleste instruktioner til Z80 afprøves. Ved øvelserne omtales desuden små kredsløb, så man kan anvende computeren til styring.
Det supplerende stof indeholder et diagram over computermodellen, en skitse af mikroprocessorens historie og en omtale af computersprog.

Hovedtekst

I mange år har man kendt automater af forskellige slags. F.eks. er veksleautomater, automatpiloter og automatiske vaskemaskiner velkendte begreber for os. En automat er et apparat konstrueret til at kunne udføre en bestemt funktion. Computeren er også en automat. Et program, der udføres, vil f.eks. få computeren til at udregne kvadratroden af et tal, styre et fabriksanlæg, spille et bip-spil, virke som et tekstbehandlingsanlæg eller styre et missil og sig selv mod "målet". En computer er således en meget generel automat, for med den samme maskine (hardware) kan man med de rette programmer (software) klare mange forskellige opgaver.
En anden del af computerens styrke ligger i, at den kan behandle alt, som kan omformes til binære koder, så som almindelige tal, bogstaver, lyd og billeder. Når disse ting først er omformet til binære koder, er behandlingen inde i computeren den samme. Det hele afvikles, når det kommer til stykket, ved, at computerens centralenhed udfører de fundamentale operationer, den er konstrueret til at kunne udføre. Det er så simple ting som addition af to binære tal og simple logiske operationer, foruden at den kan flytte rundt på bitmønstre mellem forskellige registre og hukommelsesceller i regnelageret. Det kan derfor undre, at computeren overhovedet kan have nogen betydning. Men når computeren alligevel anvendes så meget, skyldes det, at den udfører disse simple operationer med en utrolig fart - med flere millioner operationer pr. sek.'Dertil kommer, at prisen for en computer er blevet overkommelig for de fleste.



101 Skitse af en computers opbygning.

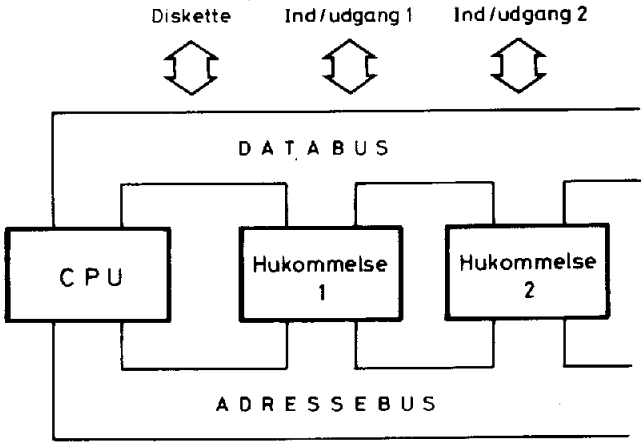
Computerens opbygning.

Som vist på figur 101 kan computeren deles op i en indlæseenhed, en centralenhed og en udlæseenhed. Indlæseenheden kan være et almindeligt skrivemaskine-tastatur, og udlæseenheden kan være en fjernsynsskærm eller en printer. Benyttes computeren til styring af maskiner, må de forbindes, så computeren både kan sende og modtage informationer. Dette foregår normalt gennem indgangs- og udgangsporte.
Forkortelsen **CPU** står for **C**entral **P**rocessing **U**nit og betegner undertiden hele centralenheden i en computer. Når man som vi ser på elektronikken, bruges betegnelsen CPU kun om den enhed, hvor selve udregningerne foregår. Centralenheden i en computer består altså af en CPU og et regnelager. I regnelageret lagres programmer og data elektronisk, og informationerne hentes over i CPU'en, efterhånden som de skal benyttes.

CPU'en indeholder en regne-logikenhed og i forbindelse hermed et antal registre til midlertidig lagring af data. Til styring af regnelogikenheden, de indre registre og resten af computeren indeholder CPU'en også en styreenhed. Styreenheden reagerer på de forskellige instruktioner, programmerne er opbygget af, og sørger på denne måde for at aktivere de rigtige dele af computeren i den rigtige rækkefølge, når et program afvikles. Sammenlagt udgør CPU'en i forbindelse med regnelageret hele hjernen i en computer, og resten af computeren er blot hjælpefunktioner for disse to enheder.
Alle computere indeholder en regne-logikenhed, et antal registre og en styreenhed, og det hele er forbundet til regnelageret med et antal ledninger. I mikrodatamaterne er hele CPU'en indbygget i en enkelt IC. Men efterhånden indbygges flere og flere logiske enheder i

samme IC, og man er allerede ved at kunne indføje hukommelser i samme IC som CPU'en. Dermed har man en hel computer i en enkelt IC - en såkaldt enkelt-chip-computer. En programmerbar lommeregner kan opfattes som en simpel computer, hvor elektronikken er samlet i en enkelt IC. De meget store computere er ligeledes opbygget af chips og med de samme funktionelle enheder som hos mikrodatamaterne, men her er CPU'ens regne-logikenhed, registrene og styreenheden sammensat af mange hundrede IC'er. En CRAY-1 er en af de helt store computere - den består af 280000 IC'er og vejer godt 5 tons.

Sammenkobling af CPU'en og regnelageret.
Mellem CPU'en, hukommelsen og computerens ind/udgangsporte (I/O-porte) er der et antal ledninger, som vi kan dele ind efter deres anvendelse.

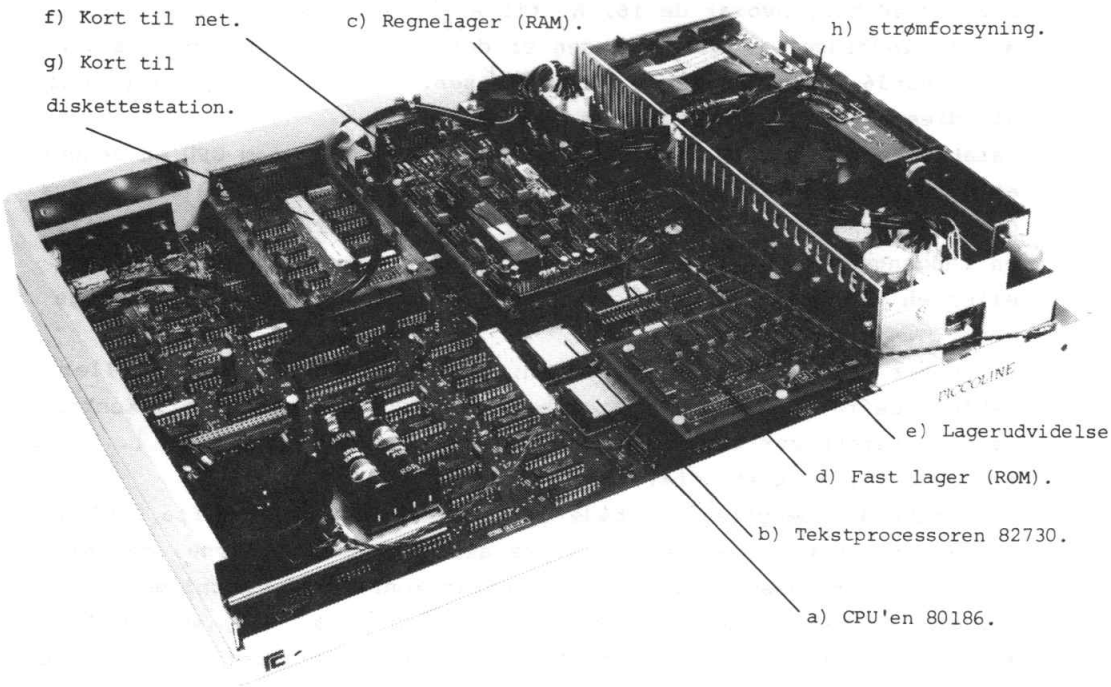


102 Kommunikationen mellem regneenheden, hukommelsen og de ydre enheder sker gennem databussen og styres med adressebussen.

Som det fremgår af figur 102, er der to sæt forbindelser fra CPU'en. Forbindelsen øverst på figuren kaldes databussen og er blot 8 parallelle ledninger (for en 8-bit maskine), der forbinder CPU'en med hukommelsen og ind/udgangene. Gennem databussen sender CPU'en data til og fra hukommelsen og ind/udgangene. Den anden forbindelse, der er vist mellem CPU'en og hukommelsen, kaldes adressebussen. Igen et antal parallelle ledninger. Gennem disse ledninger styrer CPU'en, hvor dataordene skal sendes til eller fra.

Foruden databussen og adressebussen er der et antal ledninger fra CPU'en til kontrol. Med kontrolledningerne styrer CPU'en, hvornår de enkelte komponenter koblet til busserne skal aktiveres.

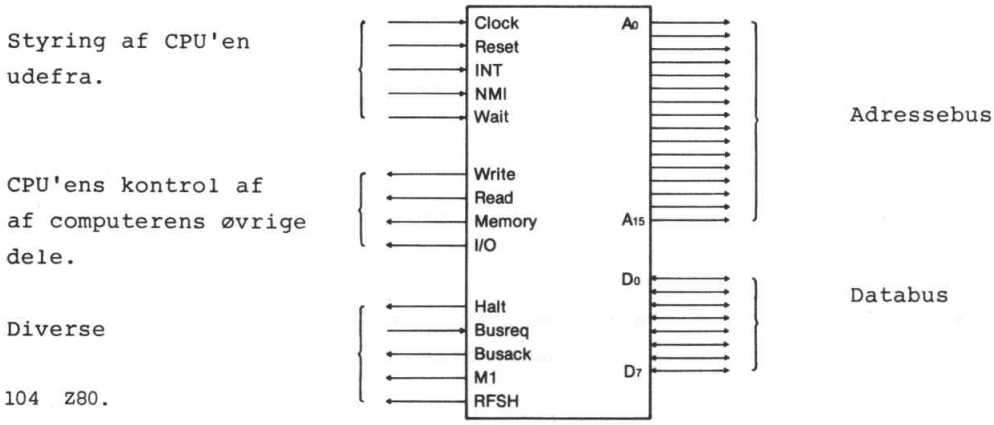
Når CPU'en udfører et program, skal det være lagret i hukommelsen i form af et antal instruktioner. Disse instruktioner fortæller CPU'en præcist, hvad den skal foretage sig under programafviklingen: hvor den skal hente tal og data, hvilke operationer, den skal udføre, hvor resultatet skal placeres, samt rækkefølgen, det hele skal udføres i. Når en enkelt instruktion er læst og udført, går CPU'en videre til den næste instruktion. En instruktion kan fylde et eller flere dataord (bytes) i hukommelsen, og det første dataord af en instruktion indeholder derfor information til CPU'en om antallet af dataord i instruktionen.
Under programafviklingen henter CPU'en instruktionerne i hukommelsen via databussen, hvorved indholdet af en adresse i hukommelsen flyttes til CPU'ens styreenhed. CPU'en vælger med adressebussen, hvor i hukommelsen de forskellige instruktioner skal hentes. Når et resultat igen skal placeres i hukommelsen, sker det ligeledes via databussen, og igen styres placeringen i hukommelsen med adressebussen.



103 Piccolineindmad.

Z80-CPU'ens benfordeling.

Ved øvelserne benyttes CPU'en Z80 udviklet af firmaet Zilog, og derfor tager vi udgangspunkt i den i det følgende. På figur 104 er vist en fortegnelse over benene på Z80 delt ind efter deres anvendelse.



104 Z80.

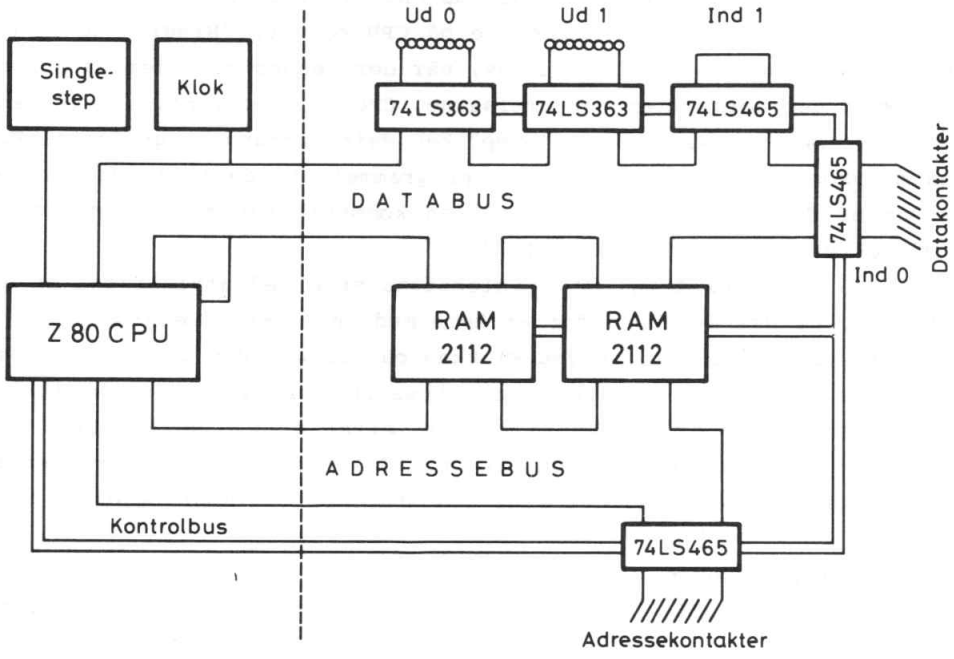
Z80 har 40 ben, hvoraf de 16, A_0 til A_{15}, benyttes til adresebussen. Med 16 ledninger i adressebussen er det muligt for CPU'en at styre 2^{16} = 65536 = 64 K adresser i hukommelsen. På figuren angiver pilene, at adresseledningerne styres af CPU'en.
Databussen kobles til de 8 ben, D_0 til D_7, hvorigennem CPU'en sender et dataord på 8 bit ad gangen. Dataordene kan sendes både til og fra CPU'en afhængigt af, om CPU'en læser dataordet fra hukommelsen eller en indgangsport, eller om CPU'en skriver dataordet til hukommelsen eller en udgangsport. Pilene markerer derfor, at data sendes begge veje.
Derudover er der dels et antal ben, som benyttes til at kontrollere CPU'en udefra, og dels et antal ben, som CPU'en benytter til kontrol af computerens øvrige komponenter (hukommelse, I/O-porte m.m.). I det følgende vil vi kort omtale nogle af disse.
Da databussen benyttes til både at sende dataord til og fra CPU'en, er det nødvendigt for den at styre alle de komponenterne, der er tilkoblet databussen, så der ikke opstår konflikter. Retningen angives med **Write** og **Read** udgangene, for når Write er Lav, svarer det til, at der skrives til hukommelsen eller en udgangsport. Er derimod Read Lav, svarer det til, at CPU'en læser data fra hukommelsen eller en indgangsport. Med **Memory**-udgangen Lav angiver CPU'en, at der kun kommunikeres med hukommelsen. Med den tilsvarende **I/O**-udgang Lav kommunikeres der kun med I/O-portene.

CPU'ens **Clock**-indgang skal have et signal fra en klok-impuls-generator. Signalet er blot et skift mellem Høj og Lav efter bestemte tidsintervaller. Klokken benyttes til at styre afviklingen af CPU'ens opgaver, så forbindelsen til computerens forskellige dele etableres i den rigtige rækkefølge og på de rigtige tidspunkter. Normalt skifter klokken mellem Høj og Lav med en fast frekvens på fra 2 til 8 MHz. Klokken styres derfor af en siliciumkrystal, da man kan få en sådan til at svinge meget præcist ved disse frekvenser.

Reset-indgangen benyttes, når man ønsker at genstarte CPU'en. Gøres Reset-indgangen Lav, resettes CPU'en, så den er klar til at starte forfra, når indgangen igen gøres Høj.

Wait-indgangen benyttes normalt, når CPU'en er koblet sammen med langsomme hukommelser eller I/O-porte. Gøres Wait-indgangen Lav, venter CPU'en med at udføre næste trin i programafviklingen.

Vores computer.



105 Diagram over vores computer.

På figur 105 er vist en principtegning af vores computer, hvor man kan se, at CPU'en Z80 er koblet sammen med hukommelsen og I/O-portene via databussen, adressebussen og nogle kontrolledninger. Vi anvender kun 8 ledninger i adressebussen og har derfor kun 2^8 = 256 adresser i hukommelsen. Som hukommelse benyttes 2 stk. 2112, der er anvendt i øvelserne kapitel 4. Den højre hukommelses-IC lagrer de 4 mindst betydende cifre af dataordene, og den venstre lagrer de 4 mest betydende cifre.

Busserne

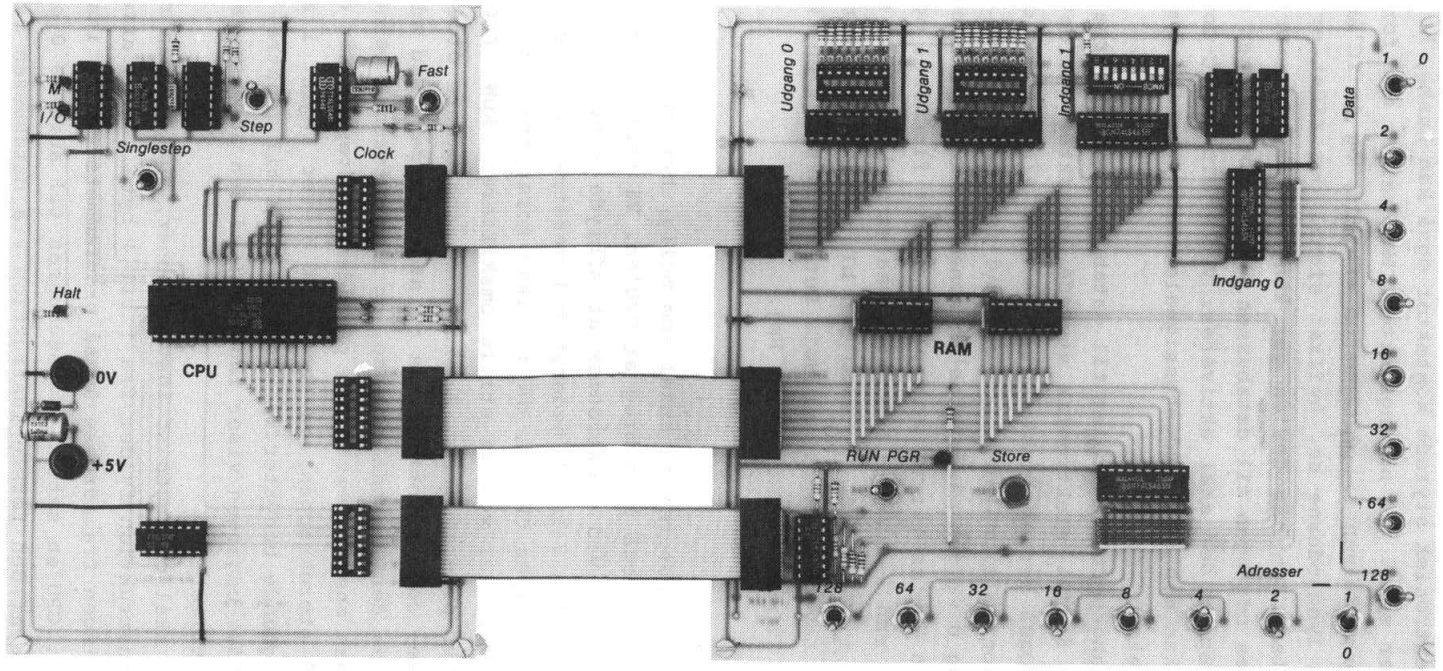
Vi har forsøgt at gøre bussernes ledningsføring så overskuelig som mulig, så man kan følge ledningerne på printpladerne. For at man skal kunne se spændingerne på busserne, er der sat en tom IC-sokkel på hver bus på CPU-kortet. Ved hjælp af et 8-bit udlæsemodul, der forbindes til disse stik med et fladkabel, kan signalerne på busserne iagttages, når CPU'en arbejder langsomt.

Singlestep og langsom klok.

Vi ønsker at kunne få computeren til at afvikle programmerne så langsomt, at vi kan følge med i, hvilke data der er på databussen, og hvilke adresser der er på adressebussen. Til det formål er der placeret lidt logik øverst til venstre på CPU-kortet, "**Singlestep**", som gør Wait-indgangen til CPU'en Lav, når der sendes signaler på busserne. Hvor det normalt er hukommelserne, CPU'en venter på, er det denne gang os. Med vippekontakten "**Step**" kan Wait-indgangen igen gøres Høj, så man kommer et "step" videre i programmet. På de to lysdioder ved siden af "Singlestep" ses, om CPU'en kommunikerer med hukommelsen, "M", eller med I/O-portene, "I/O".
For også at kunne følge med i kloksignalerne ved afviklingen af et program, er der mulighed for at køre med en klokfrekvens på ca. 1 Hz ved at stille omskifteren ved klokken på "Slow". Men da vil der komme ekstra signaler på busserne, idet busserne er Høje, når de ikke benyttes, og der er et ekstra signal, efter at CPU'en har læst den første byte af en instruktion. Det skyldes, at Z80 er konstrueret til en dynamisk hukommelse, der skal genopfriskes ca. hvert msek.

Ind- og udgangsporte.

Computeren kan kobles til omverdenen med to udgangsporte og med to indgangsporte med adresserne 0 og 1. Udgangsportene består af 8 lysdioder og er koblet til databussen med IC'erne 74LS363, der foruden



106 Billede af computeren. Computeren består af et CPU-kort med Z80 processoren og et RAM-kort med hukommelses-IC'erne 2112. Programmeringen foregår ved at stille omskifteren "RUN PGR" på PGR for ProGRammering. Programmet skal opbygges af Z80 instruktioner, der vælges med datakontakterne. Placeringen af instruktionerne i programmet vælges med adressekontakterne, og indskrivningen foregår med Store-knappen. Programmet afvikles ved at stille omskifteren RUN PGR på RUN. Udlæsningen sker på lysdioder ved de 2 udgangsporte øverst til venstre på RAM-kortet.

at være elektronisk styrede kontakter også kan huske, hvilket signal, der sidst er sendt til udgangsporten. Man kalder dem for latch. Normalt er forbindelsen til databussen afbrudt, og da IC'en kan huske, kan man på lysdioderne se, hvilket signal der sidst er sendt til udgangsporten. Kun når CPU'en gør kontrolsignalerne Write og I/O Lave, kobles udgangen til databussen. Med adressebussen styres, hvilken udgangsport der skal aktiveres. Ved hjælp af IC-soklen og et fladkabel kan man sende udgangssignalet til styring af motorer, syvsegmenter etc.
De to indgangsporte er koblet til databussen via IC'erne 74LS465, der indeholder 8 stk. elektronisk styrede kontakter (kaldet tri-state buffere). Også her er forbindelsen normalt afbrudt, og den etableres kun, når kontrolsignalerne I/O og Read fra Z80 er Lave. Signalerne til indgang 0 vælges på datakontakterne, der stilles manuelt. Indgang 1 kan enten kobles til en prøveplade med et fladkabel og styres derfra, eller man kan vælge indgangssignalet med en 8-bit mikroswitch, som passer i IC-soklen. Her skal man blot være klar over, at sluttet svarer til Lav og afbrudt til Høj. Det skyldes, at en ikke forbundet (svævende) TTL-indgang altid er Høj.

Programmering.
For at gøre computeren så enkel som mulig, har vi valgt at lade programmeringen foregå uden brug af CPU'en. Vi kobler simpelthen CPU'en fra hukommelsen ved elektronisk at afbryde kontrolsignalerne fra CPU'en med IC'en 74LS126 nederst til venstre på hukommelseskortet. Samtidigt resettes CPU'en, så den ikke forstyrrer adresse- og databusserne. Det sker ved at stille omskifteren "**RUN PGR**" på "PGR" svarende til **programmering**. Nu kan man programmere direkte i RAM'en med **adresse-** og **datakontakterne** og **storeknappen**. Med adressekontakterne vælges den ønskede adresse, og med datakontakterne vælges det dataord, man ønsker gemt i den pågældende adresse. Når det er gjort, trykkes på storeknappen, og dataordet bliver lagret i adressen. Af praktiske grunde benytter vi udgang 0 (den til venstre) som et udlæsemodul, idet udgangen viser indholdet af den adresse, adressekontakterne er stillet på.
Programmet skal opbygges af instruktioner, som Z80 kan reagere på, og en sådan instruktion fylder fra 1 til 4 byte. Z80 starter altid programafviklingen fra adresse 0, så programmets første instruktion skal starte her. CPU'en sætter adressebussen til adresse 0 og henter den første byte af den første instruktion via databussen til styreenheden, hvor den fortolkes. Er instruktionen på mere end en byte,

indeholder den første byte information om dette, og CPU'en fortsætter med at læse de resterende bytes af instruktionen.
Et program afsluttes med en "**Halt**-instruktion", der dels bevirker, at CPU'en stopper programafviklingen, og dels bevirker, at lysdioden til venstre for CPU'en lyser. Dermed kan man se, når programmet er slut, og man har en god sikkerhed for, at programmet er gennemløbet rigtigt.
Selve programafviklingen startes ved at sætte omskifteren "RUN PGR" på "RUN".

Z80 registre.
Z80 har internt i CPU'en 16 stk. 8-bit registre, som benyttes under programafviklingen. De 16 registre er delt i 8 hovedregistre og 8 alternative registre.

Hovedregistre	
A	**F**
B	**C**
D	**E**
H	**L**

De alternative registre er ligesom hovedregistrene, og man bytter om på registrene med specielle instruktioner.
Register A kaldes akkumulatoren og er det vigtigste register, som alle data til eller fra portene går igennem. Desuden benytter regneenheden for det meste akkumulatoren til lagring af data, der indgår i udregningen, og til lagring af resultatet af udregningen.
Alle 16 registre er 8-bit registre, men B og C, D og E, H og L kan kobles sammen til 3 stk. 16-bit registre.
Flagregistret F består af 8 stk 1-bit registre, der har betydning under programafviklingen. Bit 6 i flagregistret kaldes Z-registret (Z = zero) og angiver, om resultatet af en udregning er 0 eller ej. Bit 7 angiver fortegnet på resultatet, og bit 0 angiver en eventuel mente til resultatet af en udregning. Ud over de egentlige registre, som vi benytter ved programafviklingen, indeholder Z80 5 stk. 16-bit registre. Heraf er programtælleren (PC = Program Counter) det 16-bit register, der indeholder adressen til adressebussen. Når en byte af en instruktion er læst, øges programtælleren med en.
Z80 har mulighed for at indrette et særligt område af hukommelsen til midlertidigt at gemme instruktioner og data. Et sådant område kaldes en stak. I stakken henter CPU'en altid først de sidst ankomne data. Til at styre stakken er der i Z80 et register kaldet "stak-pilen", der indeholder adressen på det sidst ankomne dataord i stakken.

Øvelser

Instruktioner.

Øvelse 1. Data til og fra portene.

Z80 kan skrive og læse data direkte til og fra 256 porte med adresserne 0 til 255. De tilhørende skrive- og læseinstruktioner giver CPU'en besked om at koble porten med den rigtige adresse til databussen ved hjælp af udgangene I/O, Write, Read og ved hjælp af adressebussen.

En indlæse- og en udskriveinstruktion består af to bytes, hvor nr. 2 angiver portens adresse n.

Navn	Instruktion	Virkning
IN A,(n)	11011011 < n >	A <-- indgang n

Instruktionen flytter indholdet af indgangsporten med nummeret n til akkumulatoren. Parentesen om "n" i navnet angiver, at det er en adresse. Bemærk også, at virkningen angives fra højre mod venstre.

Navn	Instruktion	Virkning
OUT (n),A	11010011 < n >	udgang n <-- A

Indholdet af akkumulatoren flyttes til udgangen med nummeret n.

Afprøv følgende program:

Adr.	Instruktion	Navn	Virkning
0	11011011	IN A,(0)	A <-- indgang 0
1	0		
2	11010011	OUT (1),A	udgang 1 <-- A
3	1		
4	01110110	HALT	Stop

Computeren programmeres ved, at man stiller omskifteren på RAM-kortet på "PGR". Man kan nu lagre en byte af en instruktion ad gangen ved først at vælge adressen med adressekontakterne og dataordet med datakontakterne og dernæst trykke på storeknappen. Udgang 0 viser indholdet af den adresse, som adressekontakterne er sat til. På den måde kan programmet kontrolleres.
Programmet afvikles ved, at man flytter omskifteren fra "PGR" til "RUN".

Programmet skulle gerne bevirke, at computeren sender indholdet af indgang 0 til udgang 1. Skift data på datakontakterne og prøv programmet igen. Prøv også at ændre i programmet, så indholdet af indgang 1 sendes til udgang 0. Man ændrer i programmet ved at stille omskifteren på "PGR" og indskrive nye data i de adresser, der skal ændres.

Øvelse 2. Hop-instruktioner.

Z80 starter altid efter en "reset" med at afvikle programmet fra adresse 0 og fremefter, medmindre den møder en ordre om at springe i rækkefølgen.

Navn	Instruktion	Virkning
JP nm	11000011	Hop til adresse nm
	< m >	i programmet
	< n >	

JP nm får processoren til at fortsætte programafviklingen fra adressen med nummeret nm. nm er en 16-bit adresse, hvor de laveste 8 bit (m) skrives først. Instruktionen JP 07 :

11000011
00000111
00000000

vil få Z80 til at fortsætte programafviklingen fra adresse 7, uanset hvor i programmet instruktionen indføjes.

Prøv at ændre i programmet fra øvelse 1, så processoren i stedet for at standse ved Halt-ordren i adresse 4 springer tilbage i programmet og starter forfra.

Man kan f.eks. benytte tilføjelsen:

Adr.	Instruktion	Navn	Virkning
4	11000011	JP 00	Programmet
5	00000000		startes
6	00000000		forfra.

Nu vil computeren hele tiden gentage programmet, og man kan ændre på data under kørslen med datakontakterne, og vi bemærker, at udgangen skifter (næsten) samtidig.
Ofte har man brug for en betinget hop-ordre. JP Z nm får processoren til at hoppe til adresse nm, hvis resultatet af sidste udregning = 0 (Z-registret er 1). Man kan også lave andre betingede hop. F.eks. er JP NZ nm et hop til adresse nm, hvis resultatet af sidste udregning er forskellig fra 0 (Non Zero).

Øvelse 3. Addition og flytning af tal.

Flytteinstruktionen LD A,n : 00111110
< n >

flytter 8-bit tallet n til register A (akkumulatoren).

Additionsinstruktionen ADD A,n: 11000110
< n >

vil addere tallet n til indholdet af akkumulatoren og placere resultatet i denne.

Adr.	Instruktion	Navn	Virkning
0	00111110	LD A,7	A <-- tallet 7
1	00000111 = 7		
2	11000110	ADD A,6	A <-- A + 6
3	00000110 = 6		
4	11010011	OUT (1),A	Udgang 1 <-- A
5	00000001		
6	01110110	HALT	Stop

Programmet lægger forhåbentligt tallene 7 og 6 sammen og skriver resultatet til udgang 1. Ved at ændre indholdet af adresse 1 og 3 kan man få andre tal lagt sammen. Prøv det.

Ud over de allerede benyttede er der mange andre flytteinstruktioner:

```
LD A, n    : 00111110     Flytter tallet n til register A
             <   n  >

LD B, A    : 01000111     Flytter indholdet af register A til B

LD A, B    : 01111000     Flytter indholdet af register B til A

LD A,(nm)  : 00111010     Flytter indholdet af adresse nm i
             <   m  >     hukommelsen til register A
             <   n  >

LD (nm),A  : 00110010     Flytter indholdet af register A til
             <   m  >     adresse nm i hukommelsen
             <   n  >
```

Øvelse 4. Addition fra indgangene.

Adr.	Instruktion	Navn	Virkning
0	11011011	IN A,(0)	A <-- Indgang 0
1	00000000		
2	01000111	LD B,A	B <-- A
3	11011011	IN A,(1)	A <-- Indgang 1
4	00000001		
5	10000000	ADD A,B	A <-- A + B
6	11010011	OUT (1),A	Udgang 1 <-- A
7	00000001		
8	01110110	HALT	Stop

Find først ud af, hvad programmet laver, ved at gennemgå de forskellige instruktioner.
Prøv dernæst programmet.
Lav en tilføjelse til programmet, så det starter forfra, når det er kørt igennem. Dermed kan tallene ændres under kørslen.

Øvelse 5. Optælling.

Instruktionen INC A får processoren til at lægge 1 til indholdet af register A. Med en hop-ordre kan dette gentages i det uendelige.
Hvis programmet virker, vil man se lysdioderne på udgang 1 blinke. De laveste bit blinker så hurtigt, at der hele tiden er lys. Afhængigt af klok-frekvensen kan man se de højeste bit blinke langsommere.

Adr.	Instruktion	Navn	Virkning
0	00111100	INC A	A <-- A + 1
1	11010011	OUT (1),A	Udgang 1 <-- A
2	1		
3	11000011	JP 00	Hop til adres-
4	0		se 00
5	0		

Øvelse 6. Adresse- og databusserne.

Prøv at skifte til "singlestep" med kontakten til venstre på CPU-kortet. Ved "singlestep" udfører computeren kun et "step" ad gangen. Computeren arbejder med fuld hastighed og stopper, hver gang busserne aktiveres. Med 8-bit udlæsemoduler forbundet til busserne kan du prøve at "steppe" gennem programmet. Man kan på de to lysdioder til venstre for singlestep-kredsløbet se, om det er hukommelsen, eller det er I/O-portene, der kommunikeres med.
I stedet for at "steppe" gennem programmet, kan du prøve at slå kontakten ved klokken over på "Slow". Derved skifter klokken med ca. 1 Hz, og man kan følge trafikken på busserne, mens programmet afvikles. Prøv at forbinde signalet fra klokken til et udlæsemodul, så du kan følge med i processorens arbejde ved programafviklingen. Når busserne ikke benyttes af CPU'en, er de svævende - det registrerer man på udlæsemodulerne ved at alle lysdioder lyser - altså ved at alle ledninger er Høje. Desuden observerer man, efter at CPU'en har læst den første byte af en instruktion, at den tilsyneladende læser i en tilfældig adresse i hukommelsen, og at databussen er svævende. Det skyldes, at CPU'en er beregnet til en dynamisk hukommelse. Så efter at CPU'en har læst den første byte af en instruktion, sender den signal til hukommelsen om at genopfriske den adresse, adressebussen sættes til.

Øvelse 7. Optælling med 16 bit.

Z80 er internt en 16-bit maskine. Med særlige instruktioner kan man i en operation få den til at regne og flytte med 16 bit ad gangen. Ved den slags instruktioner er det underforstået, at registrene B og C slås sammen til et 16-bit register med de laveste 8 bit i register C. Ligeledes slås registrene D og E samt H og L sammen til 16-bit registre. Skal de 16 bit til eller fra CPU'en, må det ske på sædvanlig vis gennem akkumulatoren. Evt kan der kobles et oscilloskop eller en højttaler med en modstand større end 100 Ω til de forskellige "bit".

Adr.	Instruktion	Navn	Virkning
0	00000011	INC BC	BC <-- BC + 1
1	01111001	LD A,C	A <-- C
2	11010011	OUT (1),A	Udgang 1 <-- A
3	00000001		
4	01111000	LD A,B	A <-- B
5	11010011	OUT (0),A	Udgang 0 <-- A
6	00000000		
7	11000011	JP 00	Hop til adresse
8	00000000		00
9	00000000		

Prøv at forstå programmet, og prøv det af. Nu skal de to 8-bit udgange opfattes som en samlet 16-bit udgang. Prøv at iagttage strukturen i to-talsystemet ved at følge optællingen.

Med et stopur kan du tage tid på et gennemløb, og med klokken på "Slow" kan du tælle antallet af klokpulser ved et gennemløb, og derved kan klokfrekvensen beregnes. Gør det. Hvor lang tid vil en optælling til 2^{16} (et gennemløb) tage med klokken på "Slow"?

Til sammenligning kan nævnes, at en Piccoline arbejder med en frekvens på 6 MHz, og at den er en 16-bit maskine med en 16-bit databus. Vurder, hvor mange gange hurtigere en Piccoline vil gennemløbe et tilsvarende program.

Øvelse 8. Gangeprogram (255 x 256).

Når man ganger 2 med 3, svarer det til at udføre additionen 2 + 2 + 2. Dette kan vi også få computeren til at gøre. Programmet er bygget op som en uendelig sammenlægning af det samme tal a og en hop-ordre, der afbryder sammenlægningen efter b gennemløb.

Sammenlægningen finder sted efter instruktionen i adresse 10, hvor BC-registret indeholder tallet, der lægges til. Resultatet af additionen placeres i HL-registret. Sammenlægningen afsluttes efter b gange, og det sker ved, at JP Z ordren i adresse 12-14 får processoren til at springe til adresse 18, når indholdet af akkumulatoren er 0. Prøv at forstå programmet og afprøv det.

Den opmærksomme læser vil bemærke, at man kan undgå de to hop-ordrer JP Z og JP i adresserne 12-17, hvis man i stedet benytter en JP NZ hop-ordre: 11000010. Tomme pladser i programmet kan udfyldes med instruktionen NOP (No OPeration): 00000000.

Adr.	Instruktion	Navn	Virkning
0	00111110	LD A,0	
1	00000000		Nulstilling
2	01100111	LD H,A	af registrene
3	01101111	LD L,A	B, H og L
4	01000111	LD B,A	
5	11011011	IN (1),A	
6	00000001		Indgang 1 til
7	01001111	LD C,A	C og indgang 0
8	11011011	IN A,(0)	til A
9	00000000		
10	00001001	ADD HL,BC	
11	00111101	DEC A	
12	11001010	JP Z 18	Hoppe ud
13	00010010		
14	00000000		Uendelig løkke
15	11000011	JP 010	
16	00001010		
17	00000000		
18	01111100	LD A,H	
19	11010011	OUT (0),A	Udgang: H til
20	00000000		udgang 0 og L
21	01111101	LD A,L	til udgang 1
22	11010011	OUT (1),A	
23	00000001		
24	01110110	HALT	Stop

Øvelse 9. Forsinkelseskredsløb.

Ofte er det ønskeligt at kunne forsinke afviklingen af et program, så man kan følge med ved afviklingen, eller så man kan køre programmet efter bestemte tidsintervaller. Det kan klares ved at man indbygger et forsinkelsesprogram. Ønskes en lille forsinkelse, kan det gøres ved at hente input fra datakontakterne og tælle det ned til 0.

Adr.	Instruktion	Navn	Virkning
0	11011011	IN A,(0)	A <-- indgang 0
1	0		
2	00111101	DEC A	A <-- A - 1
3	11000010	JP NZ 02	Hop til adresse
4	00000010		02 hvis resulta-
5	00000000		tet ikke er 0
6	01110110	HALT	Stop

Ønskes en længere forsinkelse, kan man i stedet vælge, hvor mange gange programmet skal tælle fra 255 til 0. Programmet tæller ned i register B fra 255 til 0, og antallet af nedtællinger hentes fra datakontakterne (indgang 0), og anbringes i Akkumulatoren. Hvis tidsintervallerne er for store, kan man blot lade nedtællingen i B foregå fra et mindre tal end de 255, som er indlagt i programmets adresse 3.

Adr.	Instruktion	Navn	Virkning
0	11011011	IN A,(0)	A <-- indgang 0
1	0		
2	00000110	LD B,255	B <-- 255
3	11111111		
4	00000101	DEC B	B <-- B - 1
5	11000010	JP NZ 04	Hop til adresse
6	00000100		04 hvis resulta-
7	00000000		tet ikke er 0
8	00111101	DEC A	A <-- A - 1
9	11000010	JP NZ 02	Hop til adresse
10	00000010		02 hvis resulta-
11	00000000		tet ikke er 0
12	01110110	HALT	Stop

Prøv at benytte forsinkelsesprogrammerne til at frembringe lyde gennem en højttaler (indre modstand > 100 Ω).

Øvelse 10. Division med hele tal (langsom).

Programmet er konstrueret som et langsomt gangeprogam. I stedet for at udnytte positionssystemet tæller programmet blot, hvor mange gange tallet, der divideres med (fra indgang 1) kan trækkes fra tallet, der divideres op i (fra indgang 0). Resultatet sendes til udgang 0 og resten til udgang 1

Adr.	Instruktion	Navn	Virkning
0	11011011	IN A,(1)	A <-- indgang 1
1	1		
2	01000111	LD B,A	B <-- A
3	11011011	IN A,(0)	A <-- indgang 0
4	0		
5	00001110	LD C,0	Nulstil C
6	00000000		
7	00001100	INC C	C <-- C + 1
8	10010000	SUB B	A <-- A - B
9	11110010	JP P 07	Gentag subtraktionen
10	00000111		hvis resultatet er
11	00000000		positivt
12	00001101	DEC C	Juster resultatet
13	10000000	ADD A,B	
14	11010011	OUT (1),A	Resten til udgang 1
15	1		
16	01111001	LD A,C	A <-- C
17	11010011	OUT (0),A	Resultatet til udgang 0
18	0		
19	11000011	JP 00	Start forfra
20	00000000		
21	00000000		

Styring.

Det følgende er tænkt som et idekatalog ved større projekter, hvor man vil benytte computeren til styring af omgivelserne.

En opgave kan f.eks gå ud på at lade computeren styre opvarmningen. Med et elektrisk termometer kan man benytte computeren til at digitalisere det analoge signal, og benytte det til styring.

En anden opgave kan gå ud på at lave et program til et stopur.

En tredie opgave kan gå ud på at omsætte binære signaler til ti-talsystemet og udlæse resultatet på syvsegmenter.

Endelig kan man anvende computeren ved faldtidsmålinger. Programmet går ud på at styre en elektromagnet med udgang 1 og tælle, til der kommer et signal på indgang 1. Optællingen starter, når datakontakt nr. 0 stilles til 0.

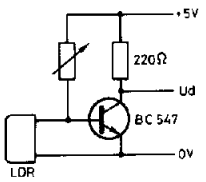
```
0    LD B,0        Nulstil B og C
2    LD C,B
3    IN A,(0)      Check datakon-
5    AND 1         takt nr. 0
7    JP NZ 03
10   LD A,1        Start fald med
12   OUT (1),A     udgang 1
14   INC BC        Optælling
15   IN A,(1)
17   AND 1         Check indgang 1
19   JP NZ 014     Gentag optælling
22   LD A,C
23   OUT (1),A     Udgang:
25   LD A,B        C til udgang 1 og
26   OUT (0),A     B til udgang 0
28   HALT          Stop
```

Indgange.

Indgange til computeren kan kobles direkte til stikket på indgang 1, eller man kan forbinde indgang 1 med et fladkabel til en 16-bens prøveplade og forbinde indgangene til printspydene på prøvepladen.

Fotocelle.

En fotocelle kan laves af en fotoresistor (LDR-modstand eller "bamseøje") og en pære koblet sammen med 2 modstande og en transistor efter følgende diagram:

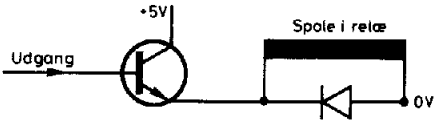


108 Fotocelle.

Udgangen er Lav, når lyset er afbrudt, og med skydemodstanden reguleres følsomheden af fotocellen. Ved at koble fotocellen til indgangen af computeren kan man med et program få computeren til at reagere på signalet fra fotocellen.

Udgange.

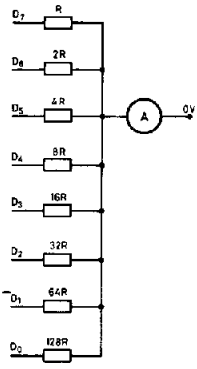
Med udgangene fra computeren kan man styre elektriske apparater så som motorer, magnetventiler, digital-til-analog- og analog-til-digital-convertere etc. I mange situationer er det dog nødvendigt at koble et relæ til udgangen. Relæerne skal kunne skifte ved ca. 3 V, og ofte kræver relæerne større strømstyrke, end IC'en 74LS363 kan levere. Derfor forstærkes signalet med en transistor, f.eks. BC547. Husk at støjdæmpe spolen på relæet ved at koble en diode over spolens poler modsat strømretningen.



109 Strømforstærkning til relæ.

Digital til Analog omsætter.

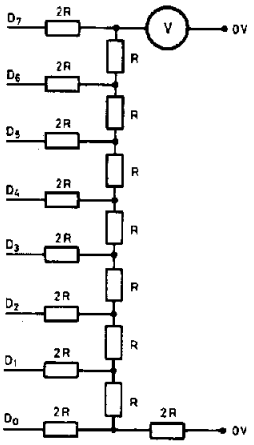
Et digitalt signal er et tal i det binære talsystem, og et analogt signal er en spænding eller en strømstyrke. En digital til analog omsætter (DA-converter) er et elektrisk kredsløb, der omformer det digitale signal til en spænding eller en strømstyrke, som er proportional med talværdien af det digitale signal.
Det simpleste kredsløb opbygges af modstande med resistanserne R, 2R, 4R,$\cdots 2^{n-1}$R. Den mindste modstand kobles til den mest betydende bit, hvor der således kommer til at løbe den største strøm. Når modstandene kobles som vist på figur 110, vil strømstyrken gennem amperemetret være proportional med det binære tal, der er sendt til udgangen D_7, $D_6, \cdots, D_0$. Ønskes i stedet et signal i form af en spænding, kan man blot erstatte amperemetret med et voltmeter, hvor den indre modstand er stor i forhold til den største modstand. I denne situation går der ingen strøm gennem voltmetret, og strømmen gennem modstandene kan gå begge veje.



110 Digital til analogkonverter.

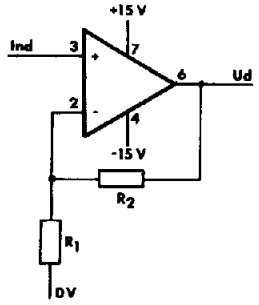
I den sidste situation vil spændingen på voltmetret variere jævnt i området Lav til Høj spænding.
Usikkerheden på den mindste modstand er helt afgørende for den samlede nøjagtighed, og derfor benytter man ofte en anden metode - den såkaldte "R-2R-kæde".

R-2R-kæden forbindes til udgangen som vist på figur 111. Igen skal voltmetrets indre modstand være stor for ikke at belaste kredsen.



111 DA-converter med "R-2R-kæde".

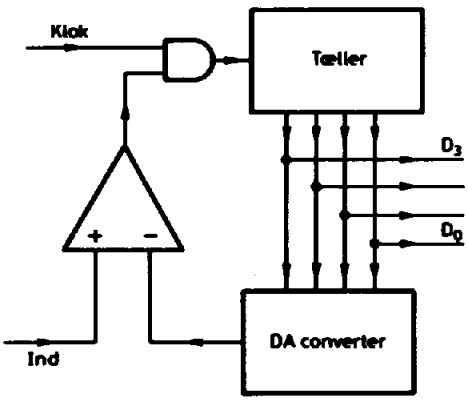
Hvis man ønsker det, kan signalet forstærkes med en operationsforstærker (f.eks. µA741 fra Philips eller TL081C fra Texas). Operationsforstærkeren leveres som en 8-bens IC. For at forstærke signalet skal operationsforstærkeren kobles, som figur 112 viser. Spændingsforstærkningen er $1+R_2/R_1$.



112 Operationsforstærker.

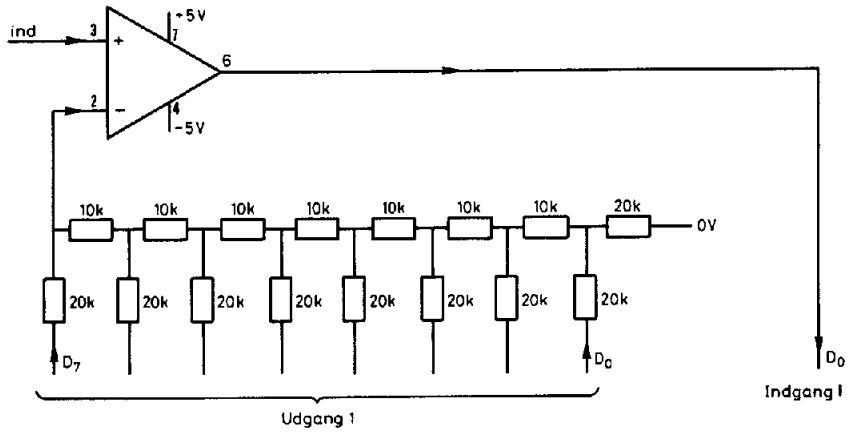
Analog til Digital omsætter.

Princippet i en analog til digital omsætter (AD-converter) er at omsætte et analogt signal til et binært tal, så talværdien svarer til spændingens størrelse. Det simpleste kredsløb er sammensat af en tæller forbundet til en DA-converter. Tælleren starter fra 0, og når den ønskede spænding er nået, stoppes optællingen, og resultatet er det binære tal. Til at sammenligne spændingen fra DA-converteren og den spænding, der skal konverteres, benyttes igen en operationsforstærker. Opstillingen er vist på figur 113.



113 Analog til digital omsætter.

Man kan også benytte computeren som tæller og koble DA-konverteren til en af udgangene. Det tilsvarende program skal så checke signalet fra operationsforstærkeren, og det kan gøres, hvis man kobler signalet til indgang 1 (f.eks. bit 0).



114 Computeren som AD-converter.

Programmet tæller op i register B og sender resultatet til udgang 1. Når den ønskede spænding er nået, stoppes optællingen, og resultatet sendes til udgang 0.

Adr	Instruktion	Navn	Virkning
0	00000110	LD B,255	"Nulstil B"
1	11111111		
2	00000100	INC B	Tæl udgangen 1 op
3	01111000	LD A,B	
4	11010011	OUT (1),A	
5	1		
6	11011011	IN A,(1)	Check indgang 1
7	1		
8	11100110	AND 1	
9	00000001		
10	11000010	JP NZ 02	Gentag fra adresse 2
11	00000010		hvis indgang 1 er Høj
12	00000000		
13	01111000	LD A,B	
14	11010011	OUT (0),A	Resultat til udgang 0
15	0		
16	11000011	JP 00	Start forfra
17	00000000		
18	00000000		

Når programmet virker, ses optællingen på udgang 1 og resultatet på udgang 0. Konverteringen sker i området 0 V til 4 V, og ønskes et mindre område, forstærkes det analoge signal, inden det konverteres. Forstærkningen sker med operationsforstærkeren koblet som vist på figur 114. Ønskes et større område, kan det analoge signal formindskes med en spændingsdeler, inden det konverteres.

Supplerende stof

Diagram over computeren.
På figur 115 er vist et diagram over computeren. Navnene på ledningerne er angivet, så man kan se deres funktion, ligesom pilene angiver, hvilken vej signalerne sendes.
Stregen over nogle af symbolerne angiver, at signalet er aktivt, når det er Lavt. F.eks. er kontrolledningen Read aktiv Lav.

Umiddelbart er det nok med én ledning til at styre læse/skrive-funktionen af hukommelsen 2112, men da Z80 er konstrueret til meget hurtige hukommelser, er det nødvendigt at tage hensyn til den tid, det tager signalerne at nå fra CPU'en til RAM'en og omvendt. Derfor har Z80 de to kontrolsignaler Write og Read. For at benytte 2112 sammen med Z80 er der et logisk kredsløb nederst på CPU-kortet i form af IC'en 74LS00 (en Low power Schottky NAND GATE).

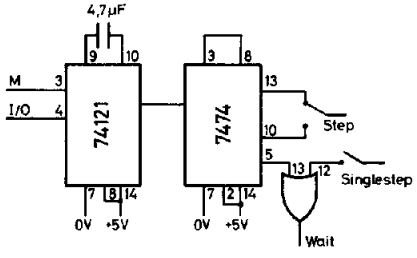
Opgave 1. Vis, at udgangen CS kun er Lav, når Memory er Lav samtidig med, at enten Read eller Write er Lave. (Husk, at en svævende indgang altid er Høj.)

Øverst til højre på computerens RAM-kort er der et logisk kredsløb, som via Z80 kontrolsignalerne og adresseledning 0 skal styre IC'erne til ind- og udgangene.

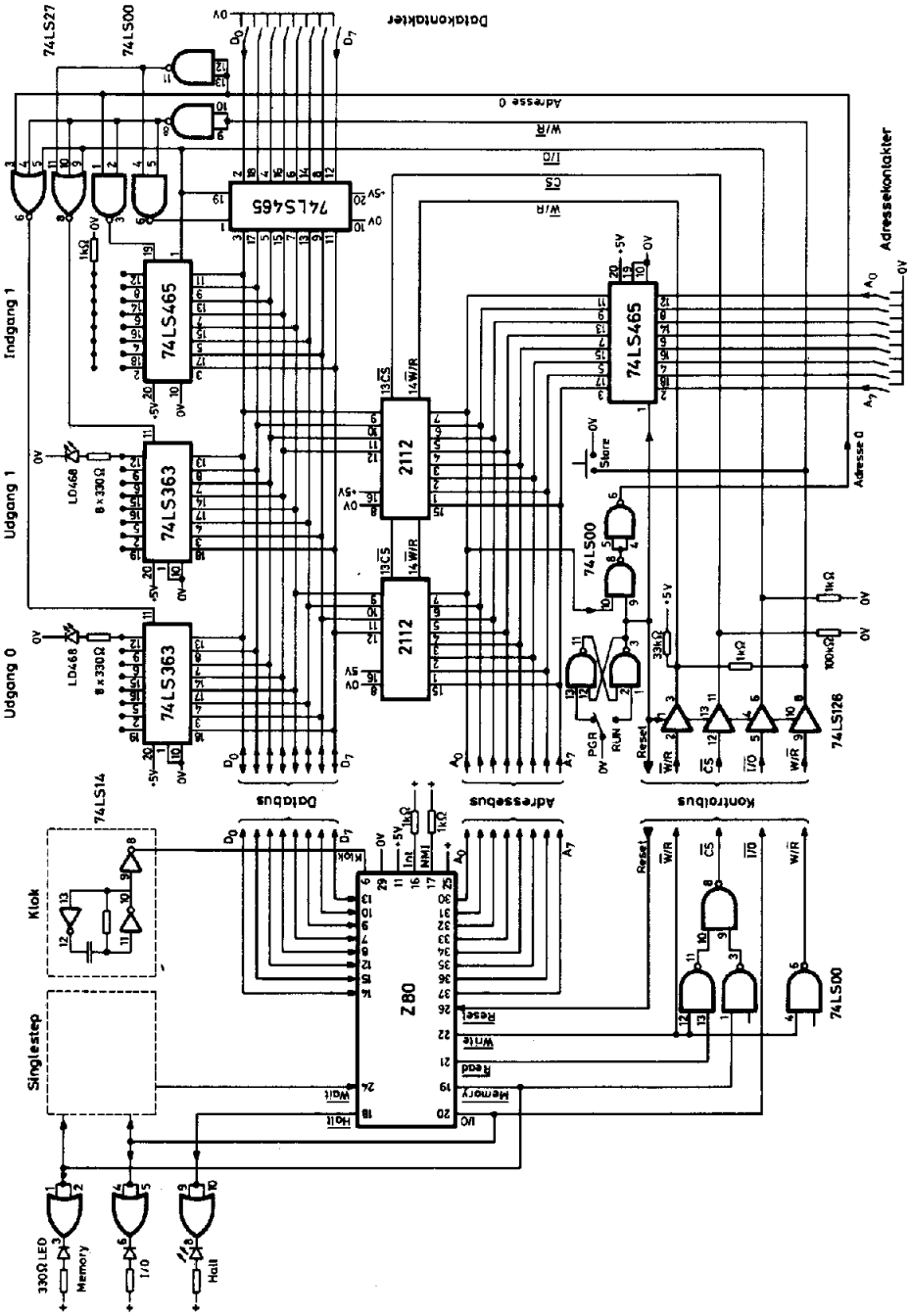
Opgave 2. Vis, at kontrolledningen til IC'en 74LS363 ved udgang 0 kun er Høj, når signalerne IO, Write og adresse 0 alle er Lave.

Opgave 3. Vis, at kontrolledningene til IC'en 74LS465 ved indgang 1 begge er Lave, når og kun når IO og Read er Lave og Adresse 0 er Høj.

Singlestep.
Kredsløbet, der kan få computeren til at "steppe", er vist på figur 116.



116 Diagram over elektronikken til Singlestep.



115 Diagram over de elektroniske kredsløb i computeren.

Mikroprocessorens historie.

Firmaet Intel Corporation konstruerede i 1971 den første mikroprocessor - en CPU i en enkelt IC. Den fik betegnelsen 4004, regnede med 4 bit ad gangen og havde indbygget 46 instruktioner. Men for at benytte den var det nødvendigt med en række hjælpekredsløb til at styre den med. Den første egentlige mikroprocessor var den tilsvarende 8 bit version fra Intel, 8008, der havde en adressebus på 14 ledninger.

Efterhånden som fremstillingsteknikken udvikledes, lavede elektronikfirmaerne større og hurtigere kredse, og som en konsekvens af dette udsendte Intel i 1973 en forbedret version af 8008 - kaldet 8080. Den er fremstillet ved en teknik, hvor man benytter MOS-transistorer af n-materiale i stedet for af p-materiale, der blev anvendt ved fremstillingen af 8008 processoren. 8080 er hurtigere og har flere instruktioner end sin forgænger, og da hele instruktionssættet fra 8008 er medtaget som en del af 8080 instruktionerne, kan man let overføre programmer udviklet til 8008 processoren.

Først i 1974 fik Intel konkurrence på mikroprocessormarkedet, idet firmaet Motorola udviklede processoren 6800. I stedet for at kopiere Intels 8080 processor valgte Motorola en opbygning kendt fra ældre computere, hvor man blandt andet styrede input og output, som om det var en del af hukommelsen - memory mapped IO. Denne teknik går igen ved processoren 6502 udsendt i 1975 af firmaet MOS Technology Incorporation, der bestod af tidligere medarbejdere fra Motorola. 6502 processoren er hurtigere og har flere instruktioner end Motorolas 6800, men til gengæld er der færre registre internt i 6502. 6502 er blandt andet kendt fra Apple II computeren, Commodore 64 m.fl.

I 1976 udsendte firmaet Zilog processoren Z80, som vi kender fra denne bog. Firmaet valgte 8080 som sit udgangspunkt, og Z80 indeholder 156 instruktioner i alt, hvori alle instruktionerne fra 8080 indgår. Antallet af 8 bit registre er fordoblet i forhold til 8080 (de alternative registre), og samtidig har man tilføjet 3 af de specielle 16 bit registre, der blandt andet benyttes ved interrupt (afbrydelse af programafviklingen). Ligesom alle de foregående processorer leveres Z80 i flere forskellige versioner med forskellige regnehastigheder. Den seneste processor, Z80 H, arbejder ved en klokfrekvens på 8 MHz.

Z80 anvendes i computere som New Brain, ZX 81, Spectrum, Piccolo, Comet, Butler m.fl., og det er den mest benyttede processor i 8 bit mikrocomputere.

IBM's 8 bit mikrocomputer, der blev introduceret så sent som i 1981, benytter Intels 8088 processor fra 1979. Det er en forbedret version af 8080 processoren, og som Z80 er det internt i CPU'en en 16 bit maskine. Desuden er antallet af registre og antallet af instruktioner forøget væsentligt i forhold til 8080.
IBM's valg af Intel's 8088 processor er i 1983/84 fulgt op med valget af processoren 80188 til firmaets 16 bit computer, der dog stadig benytter en 8 bit databus. I samme periode har IBM i øvrigt overtaget ca. 20 % af aktierne i Intel, der sammenlagt var på ca. 2 mia $ ved udgangen af 1984. Omsætningen i 1984 var på 1,6 mia $.
Af egentlige 16 bit processorer kan nævnes Motorolas MC 68000 med en 24 bit adressebus, Zilogs Z8000 med en 25 bit adressebus og Intels 8086 fra 1978 og den forbedrede version 80186 fra 1982 med 20 adresseledninger i adressebussen. Den sidstnævnte processor er den, Regnecentralen benytter i Piccolinen og i Partneren sammen med tekstprocessoren 82730. Tekstprocessoren er en hjælpeprocessor, der benyttes, når der sendes signaler til skærmen. Den styrer, hvordan teksten skal se ud på skærmen, signalerne til højgrafik, og det er også via tekstprocessoren, at man kan programmere tegnsættet.
På samme måde som tekstprocessoren 82730 er udviklet som hjælpekredsløb til CPU'en, har firmaerne udviklet utallige kredsløb, som umiddelbart kan kobles sammen med CPU'erne. Her kan nævnes matematikprocessorer udviklet specielt til addition og multiplikation af tal. En addition af 2 tal med 10-potenser går ca. 100 gange hurtigere ved brug af en matematikprocessor end ved brug af et program til en almindelig CPU. Desuden er der udviklet et stort antal IC'er til kommunikation til og fra computeren, til styring af magnetiske hukommelser, til styring af netkommunikation, til kommunikation over telefonnettet etc.

Endelig kan det nævnes, at de første 32-bit processorer er i handlen. Motorolas 68020, Zilogs Z80000 og National Semiconductors 32032 var de første, og meget snart følger Motorola med processoren 386. Foruden at arbejde med 32 bit ad gangen, er processorerne udstyret med en 32-bit adressebus, hvilket betyder, at de kan adressere en hukommelse med $4 \cdot 10^9$ adresser (12000 Mbyte).

Operativsystemet.

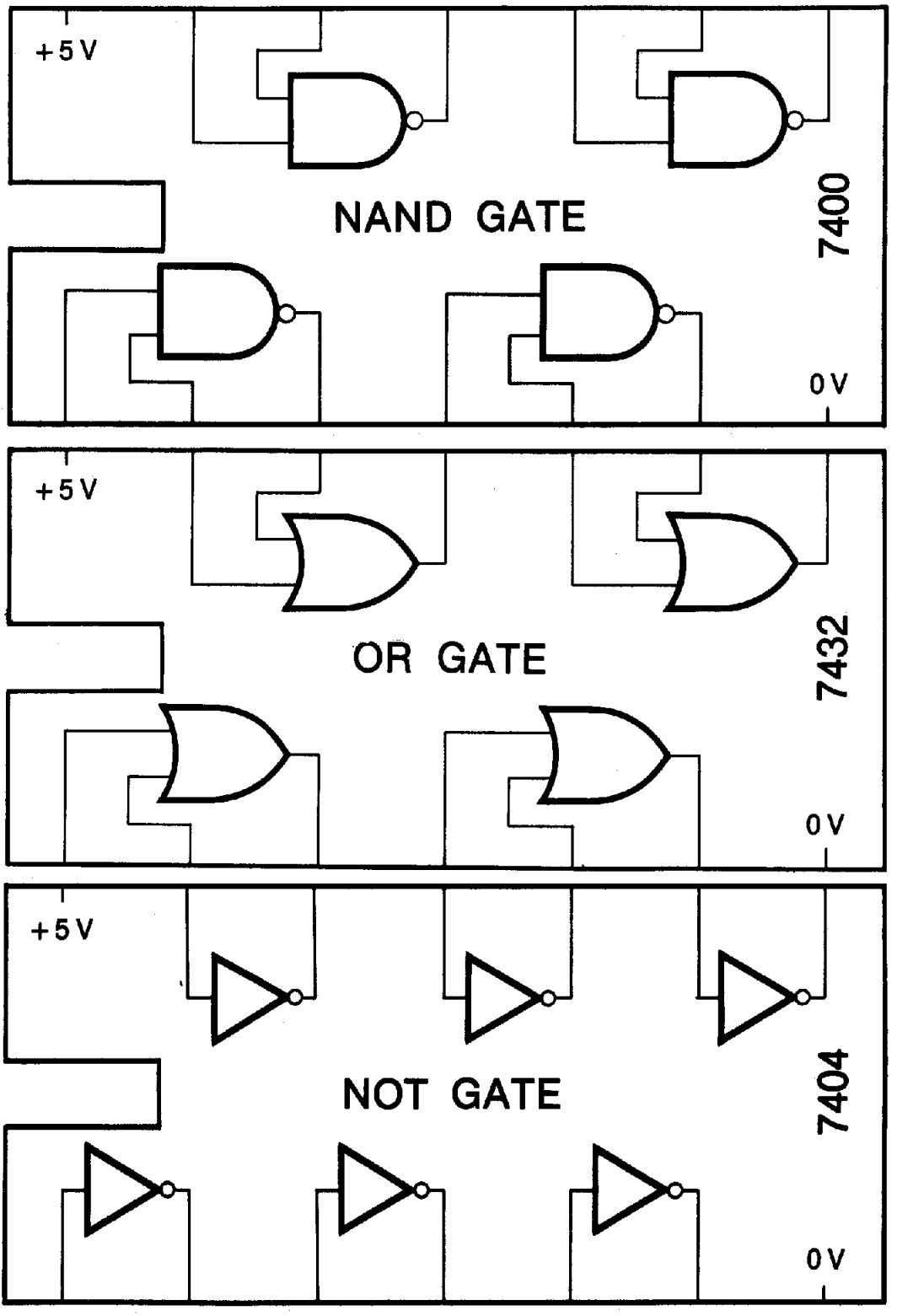
I en computer udføres mange operationer, som ikke har med afviklingen af programmer at gøre. Det kan være at hente signaler fra tastaturet, sende signaler til skærmen, flytte programmer til og fra diskettestationen, flytte programmer mellem flere disketter, starte afviklingen af et program, navngive programmer, slette programmer, styre netværkskommunikation m.m. Alt dette klares af det såkaldte operativsystem, der som det første indlæses i maskinen. Operativsystemet består af en mængde mindre programmer, der hentes frem af maskinen efterhånden, som der bliver brug for de forskellige funktioner. Det mest benyttede operativsystem til 8 bit computere er CP/M, der står for Control Program/Monitor, og som sælges af firmaet Digital Research. Der er adskillige versioner af dette operativsystem, hvor de første er beregnet til maskiner, der benytter en 8080 eller en Z80 CPU. Det seneste, CCP/M-86, er et operativsystem beregnet til 16 bit maskiner, der benytter 8086, 80186 eller tilsvarende CPU'er. Piccoline benytter CCP/M-86 operativsystemet.
For brugeren betyder operativsystemet, at man i stedet for at kende forskellige kommandoer til de forskellige dele koblet til computeren blot skal kende standardkommandoerne i operativsystemet. Så kan man behandle meddelelser til skærmen, til diskettestationen og til printeren ens. Det er så operativsystemets opgave at sende de rigtige meddelelser til den pågældende enhed i den rigtige rækkefølge og efter de regler, der gælder for denne enhed. Operativsystemet styres ved at afgive specielle kommandoer. F.eks. er "Dir" (Directory) en kommando til operativsystemet om at vise en oversigt på skærmen (eller printeren) over programmerne på diskettestationen.
Ud over alle hjælpeprogrammerne indeholder operativsystemet også en stribe anvisninger til dem, der laver sprogene til computeren. Sproget kan være Comal80, Basic, Pascal mm., og det er sproget, der bestemmer, hvordan man skal programmere computeren. Sproget er igen et program, der udnytter operativsystemets muligheder, og som følger de anvisninger, der er i forbindelse med anvendelse af operativsystemet. Fordelene ved at lade sprogene udnytte mulighederne i operativsystemet er flere. De forskellige sprog til den samme computer kan udnytte de samme grundliggende muligheder indbygget i operativsystemet, og samtidig er det i princippet muligt at overføre programmer og sprog mellem forskellige computere, når blot de benytter samme operativsystem. I praksis er der dog altid problemer med at overføre programmer mellem forskellige computere, da fabrikanterne i reglen ændrer i operativsystemet for at tilpasse det til netop

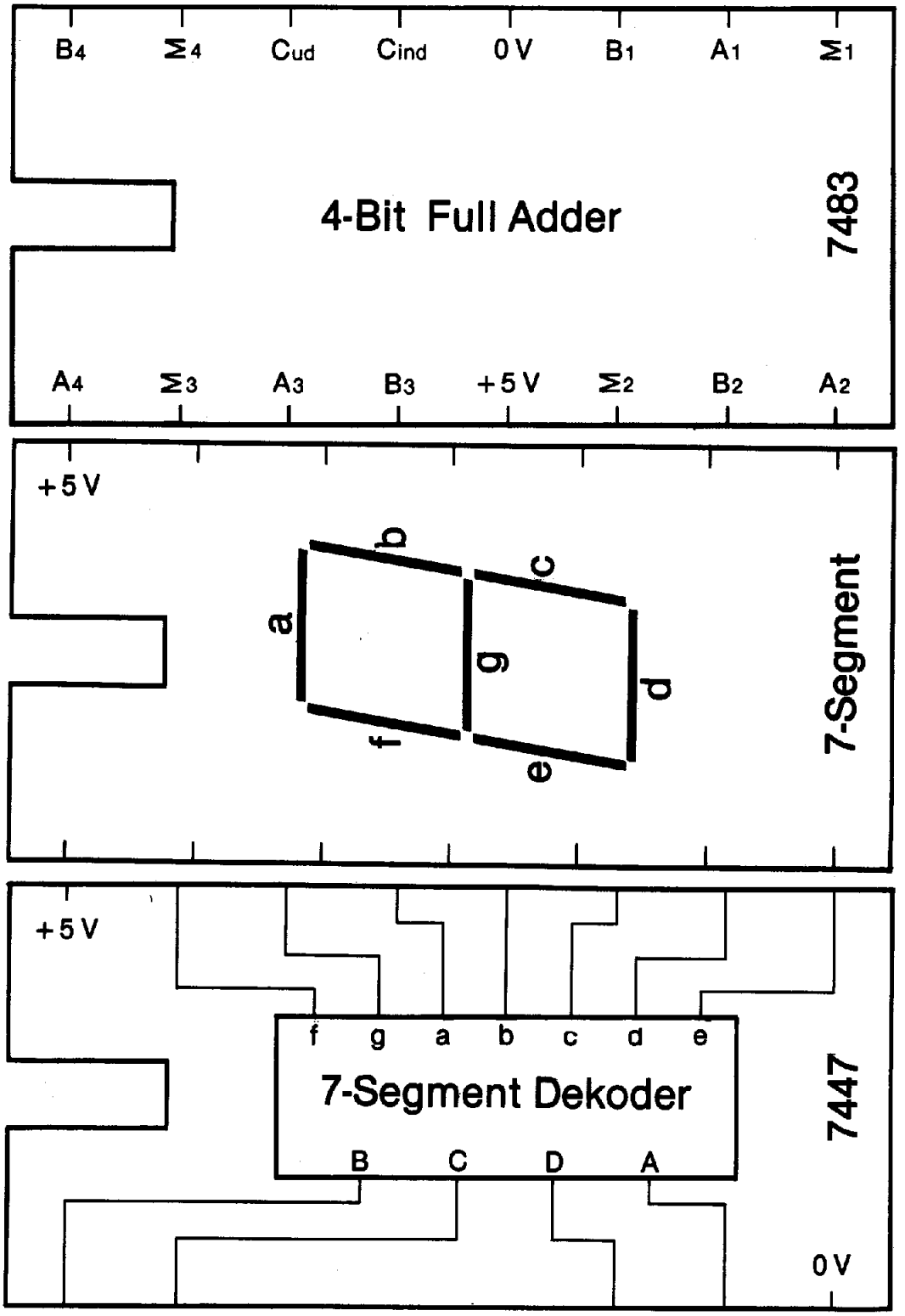
deres computer. Selv når to computere benytter forskellige operativsystemer, er det mulgt at overføre programmer mellem dem. Så skal de to sprog være nøjagtig ens, og man overfører da teksten i programmerne (kildeteksten) mellem computerne i form af ASCII-koder.
Sproget til en computer har til opgave at oversætte de ordrer og instruktioner, man bygger sit program op af, til de instruktioner, CPU'en er født med, og som den derfor kan reagere på. Det er sådanne instruktioner, vi har set på i kapitel 5 til Z80 CPU'en. Man kan skelne mellem to typer sprog. Den ene type, f.eks. Basic eller Coma180, er sprog, hvor det tilhørende program oversætter en sætning i sproget ad gangen, udfører ordren og går videre til næste sætning. På denne måde oversættes programmet til maskinkode, mens man kører programmet. Man siger, at sproget benytter en fortolker. Den anden type computersprog, f.eks. Pascal, benytter en procedure, hvor hele programmet oversættes til maskininstruktioner på en gang. Derefter kan man køre det oversatte program, og da computeren ikke skal bruge tid på at oversætte programmet til maskininstruktioner under kørslen, er programafviklingen meget hurtigere. Man siger at sproget benytter en oversætter eller en compiler.

Assembler.

På samme måde som et computersprog har til opgave at oversætte sprogets kommandoer til maskininstruktioner, kan man til de fleste computere få en "Assembler". Assembleren oversætter navnene på maskininstruktionerne til CPU'en til de tilsvarende bit-mønstre, som instruktionerne består af. Det betyder, at man blot skal indtaste (og huske) navnene på de benyttede instruktioner. F.eks. kan man skrive LD A,B i stedet for 01111000.
Når man har tastet instruktionernes navne ind i den rigtige rækkefølge og givet assembleren nogle informationer, oversættes programmet til den tilsvarende maskinkode. Normalt sker der dog to ting, idet programmet både oversættes til maskinkode og til den såkaldte "Hex-kode", hvor hver maskinkode på 8 bit blot skrives som to heksadecimale tal. Instruktionen LD A,B oversættes både til 78 og til 01111000, idet $0111_2 = 7_{16}$ og $1000_2 = 8_{16}$
Man kan også lave sit eget maskinkodeprogram uden assembler ved at skrive de tilsvarende Hex-koder efter hinanden og blot lade maskinen oversætte Hex-tallene til binære tal. Derved er instruktionerne meget lettere at indtaste end med nuller og 1-taller. Imidlertid gå man så glip af de indbyggede koder i de binære instruktioner.

Stikord


+5V
NAND GATE
7400
0V
+5V
OR GATE
7432
0V
+5V
NOT GATE
7404
0V

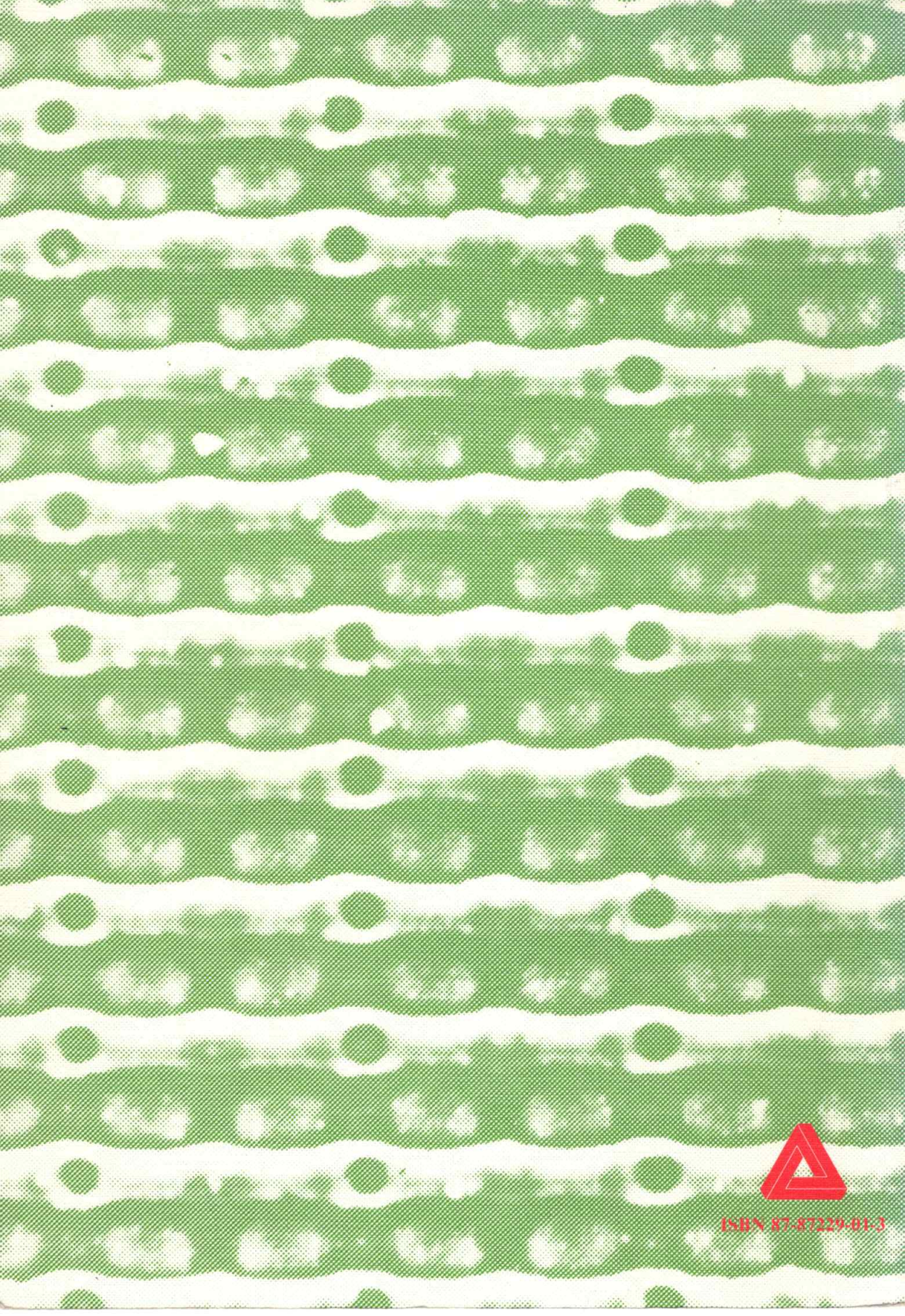

B4
Σ4
Cud
Cind
0 V
B1
A1
Σ1
4-Bit Full Adder
7483
A4
Σ3
A3
B3
+5 V
Σ2
B2
A2
+5 V
a
b
c
d
e
f
g
7-Segment
+5 V
f
g
a
b
c
d
e
7-Segment Dekoder
B
C
D
A
7447
0 V

Z80 instruktioner

I oversigten betyder r og s et af de 7
8-bit register angivet ved koden:

B=000, C=001, D=010, E=011, H=100, L=101 og A=111.

Nr	Navn	Instruktion	Virkning
1	IN A,(n)	11011011 < n >	A <-- Indgang n
2	OUT (n),A	11010011 < n >	Udgang n <-- A
3	LD B,A	01000111	B <-- A
	LD A,B	01111000	A <-- B
5	LD r,s	01 r s	r <-- s
6	LD A,n	00111110 < n >	A <-- n
7	LD r,n	00 r 110 < n >	r <-- n
8	LD A,(nm)	00111010 < m > < n >	A <-- Indholdet af adresse nummer nm
9	LD (nm),A	00110010 < m > < n >	Adresse nm <-- A
10	JP nm	11000011 < m > < n >	Hop til adresse nm
11	JP Z,nm	11001010 < m > < n >	Hop til adresse nm hvis resultatet af sidste udregning gav nul.
12	JP NZ,nm	11000010 < m > < n >	Hop til adresse nm hvis resultatet af sidste udregning ikke gav nul.
13	JP P,nm	11110010 < m > < n >	Hop til adresse nm hvis resultatet af sidste udregning var positivt.
14	ADD A,n	11000110 < n >	A <-- A+n
15	SUB A,n	11010110 < n >	A <-- A-n
16	AND n	11100110 < n >	A <-- A n
17	OR n	11110110 < n >	A <-- A n
18	ADD A,B	10000000	A <-- A+B
19	ADD A,r	10000 r	A <-- A+r
20	SUB A,B	10010000	A <-- A-B
21	SUB A,r	10010 r	A <-- A-r
22	AND B	10100000	A <-- A B
23	AND r	10100 r	A <-- A r
24	OR B	10110000	A <-- A B
25	OR r	10110 r	A <-- A r
26	INC A	00111100	A <-- A+1
27	INC r	00 r 100	r <-- r+1
28	DEC A	00111101	A <-- A-1
29	DEC r	00 r 101	r <-- r-1
30	INC HL	00100011	HL <-- HL+1
31	INC BC	00000011	BC <-- BC+1
32	DEC HL	00101011	HL <-- HL-1
33	DEC BC	00001011	BC <-- BC-1
34	SL r	11001011 00100 r	Skift register r en plads til venstre (r+r)
35	SR r	11001011 00111 r	Skift register r en plads til højre
36	NOP	00000000	No operation
37	HALT	01110110	Stop programafviklingen



ISBN 87-87229-01-3